Szenenabhängige Online-Adaption von Manipulationssequenzen für einen Serviceroboter

zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

von der Fakultät für Informatik
des Karlsruher Instituts für Technologie (KIT)

genehmigte

Dissertation
von

Steffen Wilhelm Rühl
aus Wesel

Tag der mündlichen Prüfung	:	13. Februar 2015
Erster Gutachter	:	Prof. Dr.-Ing. Rüdiger Dillmann
Zweiter Gutachter	:	Prof. Dr.-Ing. Jianwei Zhang

Ich versichere wahrheitsgemäß, die Dissertation bis auf die dort angegebenen Hilfen selbstständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und als kenntlich gemacht zu haben, was aus Arbeiten anderer und eigenen Veröffentlichungen unverändert oder mit Änderungen entnommen wurde.

Karlsruhe, im November 2014 *Steffen Wilhelm Rühl*

Kurzfassung

Um in alltäglichen Umgebungen agieren zu können, muss ein Serviceroboter in der Lage sein, seine Handlung zielgerichtet an neue Szenen anzupassen. In dieser Arbeit werden dazu Handlungspläne zur Manipulation durch symbolische Planung erzeugt. Damit diese konsistent mit der realen Umgebung sind, wird aufgrund von sensorischer Wahrnehmung ein symbolisches Modell der Umgebung erzeugt. Dieses erlaubt es dem Roboter, die Effekte einer Aktion, bzw. Aktionsfolge auf seine Umwelt vorherzusagen. Das Modell berücksichtigt dafür an Aktionen beteiligte Objekte und unbeteiligte Hindernisse. Durch einen Simulationsschritt können Effekte auch auf unbeteiligte Objekte berücksichtigt werden. Eine Abtastung der Erreichbarkeit des Arbeitsraumes stellt die Ausführbarkeit des symbolischen Plans durch die Roboterkinematik sicher.

Aufgrund der Abhängigkeit von der Wahrnehmung der aktuellen Umgebung werden die Verfahren auf Echtzeitfähigkeit ausgelegt. Methoden der Pfadplanung und inversen Kinamtik erlauben die Rückabbildung des symbolischen Plans in die kontinuierliche Ausführungsdomäne. Die Integration eines Überwachungsverfahrens sichert eine mit dem erzeugten Plan konsistente Ausführung und die Erkennung von Ausführungsfehlern in Echtzeit. Die Ergebnisse der Arbeit wurden in Experimenten an einem robotischen Demonstrator evaluiert.

Danksagung

Die vorliegende Arbeit entstand in den Jahren 2007 bis 2014 während meiner Tätigkeit als wissenschaftlicher Mitarbeiter in der Abteilung Interaktive Diagnose- und Servicesysteme am Forschungszentrum Informatik in Karlsruhe. Herrn Prof. Dr.-Ing. Rüdiger Dillmann danke ich besonders für die Anregung zu dieser Arbeit, die wissenschaftliche Förderung, die stets vorhandene Diskussionsbereitschaft und für die Übernahme des Hauptreferates. Für die freundliche Übernahme des Korreferates gebührt mein ganz besonderer Dank Herrn Dr.-Ing. Jianwei Zhang vom Institut TAMS (Technical Aspects of Multimodal Systems) der Universität Hamburg.

Ein besonders großer Dank gebührt Zhixing Xue für die spät-abendliche Diskussionen, die den Weg dieser Arbeit maßgeblich beeinflusst haben und für seinen unermüdlichen Einsatz bei der Pflege unserer Roboter. Für viele wertvolle Gespräche und Hinweise, die zum Gelingen dieser Arbeit beigetragen haben, danke ich J. Marius Zöllner und Sven Schmidt-Rohr. Thilo Kerscher und Arne Rönnau danke ich dafür, dass sie als Abteilungsleiter die Freiräume zur Promotion in der Gruppe IDS geschaffen haben.

Zu dem für die Experimente verwendeten Robotersystem haben Andreas Hermann, Rainer Jäckel, Sven Schimdt-Rohr und Zhixing Xue unersetzliche Beträge geleistet. Ebenso eine Reihe von Studenten: Pascal Becker, Gerhard Dirschl, Heiko Donat, Klaus Fischnaller, Andreas Konle, Dipika Kumar, Johannes Mangler, Sebastian Mendez, Sebastian Strzeszewski, Johannes Steudle, Marco Vetter, Wenlei Wu, Yi Xie und indirekt noch viele mehr. Euch allen einen großen Dank, ohne eure Arbeit gäbe es keine erfolgreichen Versuche in dieser Dissertation zu beschreiben.

Für ihre Hilfe als Korrekturleser bedanke ich mich herzlich bei Andrea Scheiwe, Ann Katrin Rühl, Hans-Wilhem Rühl, Johanna Kraft, Kirstin Schamm, Hanna Schubert, Paula Menzel, Sonja Göttl und Thomas Schamm.

Meinen aktuellen und ehemaligen Kollegen in den Gruppen IDS und TKS danke ich für die beste vorstellbare Arbeitsatmosphäre, die angeregten Gespräche beim Mittagessen und nach Feierabend sowie die hervorragende Zusammenarbeit. Meinen Büronachbarn Andreas Hermann und Marc Zofka gilt darüber hinaus der Dank für den gut funktionierenden Austausch von Doppelkeksen und anderen Grundnahrungsmitteln. Ganz besonders Danken möchte ich Tho-

mas Schamm für seine langjährige Freundschaft und mittägliche Joggingrunden sowie Andreas Hermann für die alljährlichen Ausflüge nach Budapest.

Am Ende des Tages stets ein offenes Ohr für die Sorgen des Tages hatte die „Schweiwe Familie" Andrea Scheiwe, Hanna Schubert, Joachim Fischer und Henning Scheiwe. Ohne euch wäre dieses Blatt noch immer weiß und ich verzweifelt, dafür gilt euch mein Dank.

Mein größter Dank geht an meine Eltern Annemarie und Hans-Wilhelm Rühl, die mit Lego und Elektronikbaukästen meine Begeisterung für Technik gefördert und mich nicht nur während der gesamten Ausbildung immer unterstützt haben. Danke für die Liebe, den Rückhalt und die Anstöße zur richtigen Zeit. Das Gleiche gilt auch für meine Schwester Ann Katrin.

Karlsruhe, im November 2014 *Steffen Wilhelm Rühl*

Inhaltsverzeichnis

1. Einleitung

Bei dem Wort „Roboter" denken viele Menschen an elektronische Übermenschen auf zwei Beinen mit Kopf, Armen und eigenem Willen. So liefert etwa eine Bildersuche nach dem Wort „Roboter" bei Google das in Abb. 1.1 abgedruckt Ergebnis. Es beinhaltet mehrheitlich Bilder menschenähnlicher Systeme, was darauf schließen lässt, dass diese im Internet besonders häufig zusammen mit dem Schlüsselwort „Roboter" vorkommen. Dies spiegelt in keiner Weise die Verbreitung von Robotern in der Wirklichkeit wieder; stattdessen ist es ein Produkt unserer Wahrnehmung von Robotern. Diese ist geprägt von einem Roboterbild aus Science Fiction Literatur und Filmen. Ein dort häufig wiederkehrendes Motiv ist die Maschine, die sich gegen ihren Schöpfer wendet. Und um hierzu in der Lage zu sein, werden Bilder von Robotern aufgebaut, die dem Menschen in intellektueller sowie körperlicher Hinsicht weit überlegen sind. Ein bekanntes Beispiel für einen Roboter in einem solchen Plot ist der T1000 aus dem Film „Terminator 2: Judgment Day" aus dem Jahr 1991, in dem die Maschinen der Zukunft in einen Krieg gegen die Menschheit ziehen.

Aber die Roboter des Science Fiction Genres müssen nicht zwingend böse sein, auch das entgegengesetzte Thema, Empathie mit der Maschine, führt zu menschlichen Robotern. Im Film „A.I. – Artificial Intelligence" von 2001 wird ein Roboter mit dem Erscheinungsbild eines Kindes zum emotionalen Ersatz für ein Kind. Der Roboter wird von einem Schauspieler gespielt; er ist perfekt menschlich dargestellt. Der Film befasst sich mit den emotionalen Beziehungen zwischen dem Roboter und den Menschen in seiner Umwelt.

Heute tatsächlich im Einsatz befindliche Roboter haben mit dem beschriebenen Roboterbild kaum Ähnlichkeiten. Bei den meisten Systemen handelt es sich um Manipulatorarme mit sechs Gelenken und einem spezialisierten Werkzeug. Sie werden in extrem strukturierter Umgebung eingesetzt, sodass sie ihre Aufgabe durch Wiederholen ein und derselben Bewegung erfolgreich erfüllen können. Sie sind weder menschenähnlich noch besitzen sie ausgeprägte Intelligenz, Bewusstsein oder gar einen eigenen Willen.

Der Einsatz von Robotern in unstrukturierten Umgebungen entwickelt sich immer weiter. So wird seit Jahren an Servicerobotern als Haushaltsassistenten geforscht. Dort sind die repetitiven Ansätze der Industrierobotik nicht anwendbar. Deshalb gibt es gute Gründe, die Entwicklung in Richtung des menschlichen Roboterbildes voranzutreiben. Eine menschliche „Anatomie" vereinfacht es dem Roboter in einer für den Menschen gemachten Umgebung zu agieren. Beine

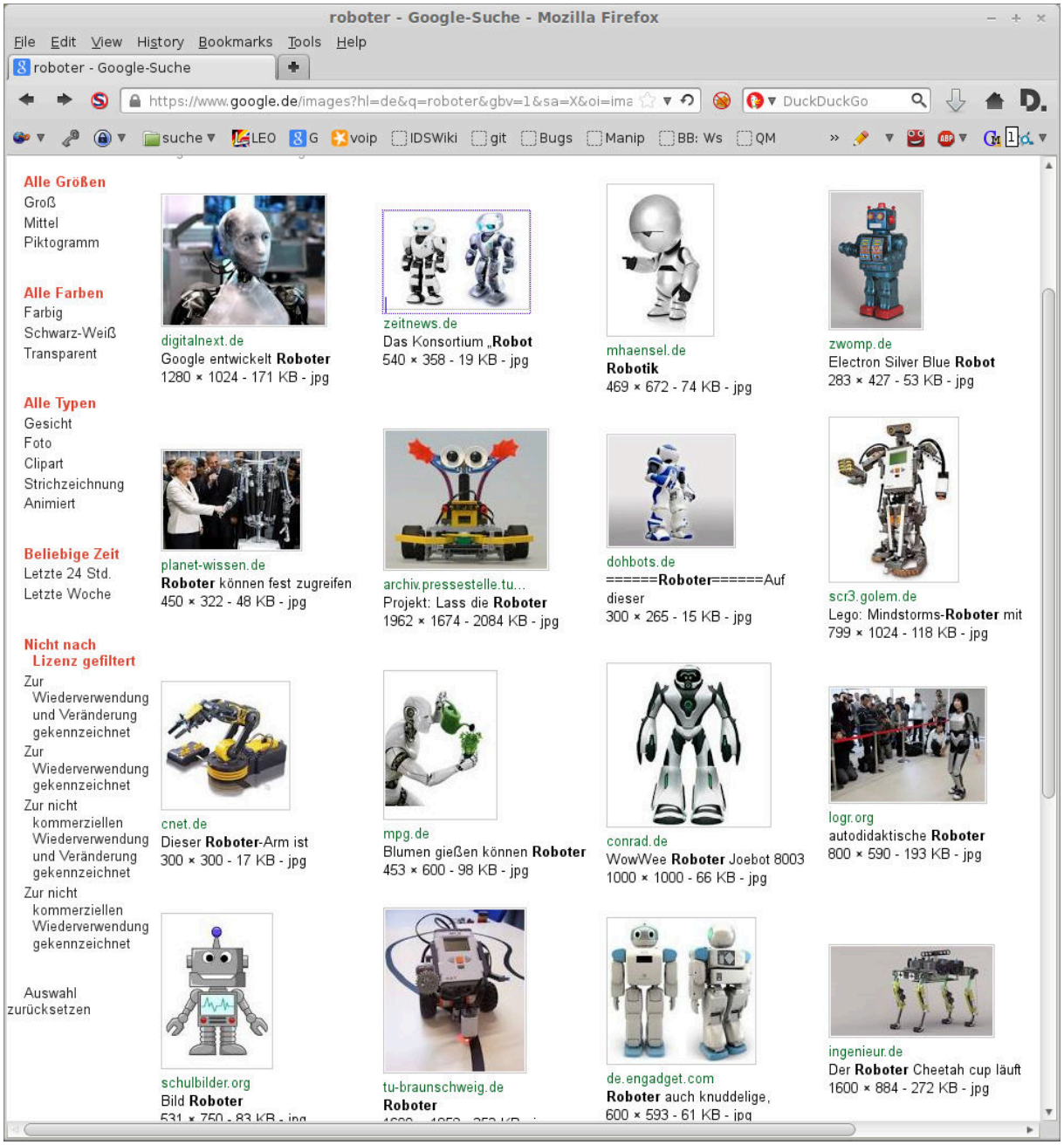


Abb. 1.1.: Ergebnis der Google Bildersuche nach dem Begriff „Roboter”. Von den sechzehn dargestellten Treffern stellen elf menschenähnlich Roboter mit Armen, Beinen, Kopf und Augen dar. Bei den meisten davon dürfte es sich um Computervisualisierungen oder funktionslose Modelle handeln.

erlauben es ihm Treppen zu steigen; mit einer menschenähnlichen Größe kann er Objekte auf einem Tisch greifen oder eine Tür öffnen. Dies sind Fähigkeiten, die kleinen Staubsaugerrobotern auf dem Boden verwehrt bleiben.

Noch wichtiger ist jedoch der Bereich der Intelligenz. In einer veränderlichen unbekannten Umgebung muss der Roboter in der Lage sein, seine Handlung zielgerichtet anzupassen, um erfolgreich mit der Umgebung interagieren zu können. So kann eine autonom geplante und ausgeführte Aktion zum Türöffnen dem Roboter anders unerreichbare Arbeitsräume eröffnen. In diesem Bereich, unter besonderer Berücksichtigung der Manipulation, forscht diese Arbeit.

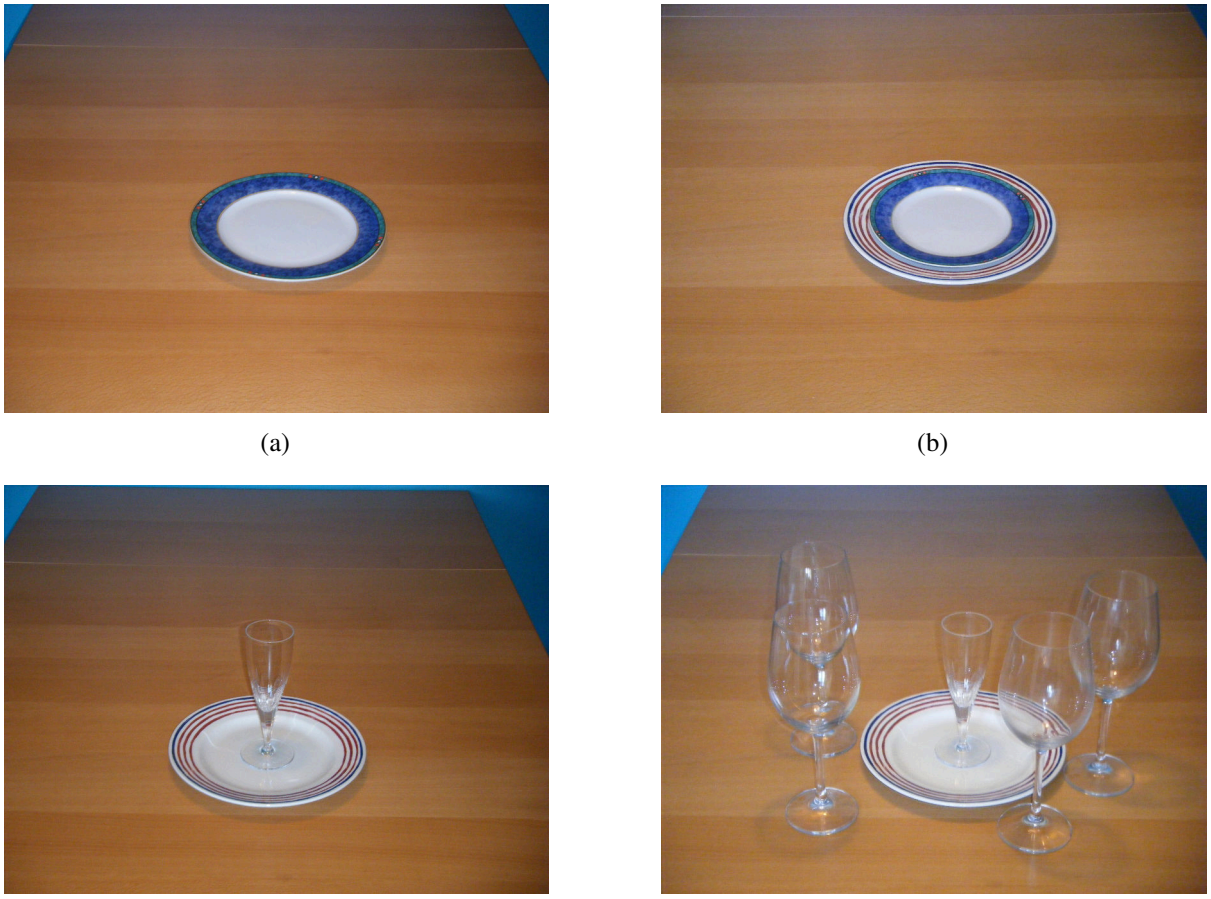
1.1. Problemstellung

Beim Einsatz eines Roboters in der unstrukturierten Haushaltsumgebung können sich (im Gegensatz zur strukturierten industriellen Umgebung) die Randbedinungen der Ausführung jederzeit ändern. Von einem Serviceroboter wird daher erwartet, dass er flexibel ist. Das bedeutet, dass der Roboter in der Lage ist, seine Handlung im Sinne des Erfüllens der Aufgabe an diese veränderten Aspekte anzupassen und sich die im Folgenden aufgeführten Aspekte verändern können:

Roboterpose Verändert sich die Roboterpose, so führt eine gegebene Konfiguration zu einer veränderten Endeffektorpose. Ist ein Serviceroboter mobil, etwa durch eine Plattform mit Rädern oder durch Beine, so tritt das Problem durch Bewegung grundsätzlich auf, da die Genauigkeit, mit der eine gewünschte Pose des Roboters erreicht wird, begrenzt ist. Insbesondere die Genauigkeit geringer als die angestrebte Genauigkeit des Endeffektors. Damit kann nicht angenommen werden, dass eine Endefektorpose durch Abfahren von Konfigurationen wiederholbar erreicht wird.

Umgebungsstruktur Die Umgebungsstruktur definiert die für eine Aufgabe relevanten Orte. Diese können etwa angeben, wo ein Schalter zu finden ist oder wo ein Objekt aufzunehmen oder abzulegen ist. Im industriellen Umfeld ist die Umgebungsstruktur fest vorgegeben, eine Änderung ist meist mit einer Änderung der Aufgabe verbunden. Für einen Serviceroboter kann so eine starre Vorgabe dagegen nicht angenommen werden. Die Aufgabe „Räume den Tisch auf" könnte sowohl auf den Wohnzimmertisch als auch auf den Esstisch angewendet werden. Im ersten Fall müssten Objekte ins Wohnzimmerregal geräumt werden, im zweiten in die Spülmaschine.

Objekte, Objektposen Unterschiedliche Objekte führen zu unterschiedlichen Griffen und Anrückkonfigurationen. Der Roboter muss seine Ausführung entsprechend anpassen.



(a) (b)

(c) (d)

Abb. 1.2.: Szene, in der eine Aufgabe ausgeführt werden soll. In (a) ist die Zielkonfiguration dargestellt; der blaue Teller steht auf dem Tisch. In (b) bis (d) entstehen durch Hindernisse zusätzliche Anforderungen an die Handlung des Roboters.

Hindernisse Objekte, die nicht mit der aktuellen Aufgabe in Verbindung stehen, sich jedoch innerhalb der manipulierten Szene befinden, werden als Hindernisse bezeichnet. Der Roboter muss Kollisionen mit diesen vermeiden.

Beziehungen zwischen Objekten Beziehungen zwischen Objekten bestimmen die Effekte von Aktionen mit Objekten auf andere Objekte. So kann ein Tablett nicht bewegt werden, ohne die Objekte darauf zu beeinflussen.

Die Summe aus Objekten und Hindernissen wird in dieser Arbeit als *Szene* bezeichnet. Sie macht Anpassungen in der Aufgabenausführung auf unterschiedlichen Abstraktionsebenen nötig. In einfachen Fällen müssen lediglich Konfigurationen des Roboters angepasst werden, um veränderten Posen gerecht zu werden.

Im Beispiel des Tischdeckens genügt das aber nicht. Dies soll an den Situationen in Abb. 1.2 erläutert werden. Die Aufgabe ist es, eine Situation wie in Abb. 1.2(a) zu erzeugen, in welcher der blaue Teller auf dem Tisch steht. Im Falle des leeren Tisches kann dies durch das Abstellen

des Tellers erreicht werden. Steht an der Zielposition bereits ein Objekt, wie in Abb. 1.2(b), so muss zuerst eine Aktion erzeugt werden, die dieses Hindernis aus dem Weg räumt. Was aber passiert mit dem oberen Teller in Abb. 1.2(b)? Es muss modelliert sein, dass sich dieser mit dem unteren Teller mit bewegt. Andererseits kann das gemeinsame Bewegen in einer Situation wie in Abb. 1.2(c) unerwünscht sein – dort besteht die Gefahr, dass das obere Objekt fällt und beschädigt wird. Der Roboter muss in der Lage sein, diese Szenen zu unterscheiden und seine Handlungen entsprechend anzupassen. Eine weitere Komplexitätsstufe kommt in Abb. 1.2(d) hinzu. Hier müssen zusätzlich Hindernisse entfernt werden, die es verhindern, das Zielobjekt zu manipulieren.

Hat ein Roboter die Fähigkeiten, um seine Handlung an Situationen wie im vorigen Beispiel anzupassen, so profitiert er davon auf mehreren Ebenen. Er kann Objekte manipulieren, die sonst für ihn unerreichbar sind. Allgemein gewinnt die Robustheit seiner Ausführung, da er einen strukturierten Arbeitsplatz erzeugen kann. Fände im letzten Beispiel ein Bahnplanner eine Lösung, den Teller zwischen den Gläsern zu greifen, so wäre diese Lösung immer noch mit der Gefahr verbunden, aufgrund von Wahrnehmungsfehlern und dem engen Raum mit einem Glas zu kollidieren und dieses zu beschädigen. Ist der Roboter in der Lage, die Szene anzupassen (das Glas also zur Seite zu stellen), können solche Fehler zuverlässig vermieden werden. Weiter gewinnt die Effizienz der Handlungsausführung, wenn der Roboter die Fähigkeit hat, Synergien bei der gleichzeitigen Manipulation mehrerer Objekte zu nutzen.

1.2. Zielsetzung und Beitrag der Arbeit

Für den Serviceroboter ist es nicht akzeptabel, die im vorigen Abschnitt diskutierten Anpassungen am Handlungsablauf durch einen Menschen durchführen zu lassen. Der Aufwand stünde in keinem sinnvollen Verhältnis zum Nutzen des Roboters. De Roboter muss daher in der Lage sein, die Ausführung einer Aufgabe autonom an die Szene anzupassen.

Ziel dieser Arbeit ist ein System, das es einem Serviceroboter ermöglicht, Aktionen in unterschiedlichen Szenen auszuführen. Dabei liegt der Fokus auf dem Erstellen zusätzlicher Aktionen, welche die Szene verändern, um die Ausführbarkeit einer angestrebten Aktion bzw. Aktionssequenz zu erreichen. Das Ergebnis ist ein Plan aus konkreten Aktionen des Roboters, die durch Trajektorien bestimmt sind. Das System muss in der Lage sein, auf Abweichungen der Ausführung mit einer Anpassung des Plans zu reagieren. Aufgrund der Abhängigkeit von der Szene muss die Generierung online, also kurz vor der Ausführung geschehen. Daher wird ein symbolisches Modell aus der Szene erzeugt, das eine zielgerichtete Planung ermöglicht. Die Arbeit beschränkt sich auf Aufgabenstellung der Manipulation. Ebenfalls wird die Mobilität des Roboters ausgeklammert.

Dabei existiert auf der einen Seite die symbolische, abstrakten Darstellung eines abstrakten Ziels bzw. Planes. Sie ermöglicht es dem Roboter, Schlussfolgerungen über die Handlungsalternativen und ihre Konsequenzen zu ziehen. Mehrere Aktionen können damit als zielgerichtete Sequenzen geplant werden. Auf der anderen Seite existiert die reale Welt als Domäne der Ausführung eines konkreten Planes. Sie ist ein zeitlich und räumlich kontinuierlicher Raum. Die Herausforderung dieser Arbeit ist es, diese beiden Domänen zusammen zu bringen und insbesondere aus dem Symbolischen heraus einen Plan zu generieren, der im Konkreten seine Bedeutung behält.

In dieser Arbeit wird dazu ein automatisches System zur Adaption von Manipulationssequenzen an eine Szene entwickelt. Der Schwerpunkt des Systems ist die Fähigkeit, abhängig von der Szene neue Aktionen zu generieren. Dazu wird ein symbolischer Planer mit Manipulationsplanern kombiniert. Unterschiedliche, im folgenden aufgelistete Komponenten werden entwickelt, um die Lücke zwischen der symbolischen Domäne des symbolischen Planers und der kontinuierlichen Domäne der Manipulationsplaner bzw. der Ausführung zu überbrücken:

Handlungsadaption Zur Plangenerierung wird eine Anbindung der Manipulationsplaner an einen symbolischen Planer definiert. Das System wird in die Robotersteuerung integriert.

Symbolisches Modell In der symbolischen Darstellung wird eine Darstellung des Arbeitsraums benötigt, um z.B. mögliche Ablageposen darzustellen. Dazu wird eine Bewertungsfunktion für Posen vorgeschlagen, um eine Selektion einer endlichen Menge geeigneter Posen durchzuführen.

Mechanische Zusammenhänge, z.B. Tablett trägt Tasse, sind für die Planung von Manipulationsaktionen bedeutend, da sie mitbestimmen, in welcher Reihenfolge Objekte manipuliert werden können. Diese Arbeit schlägt ein geeignetes symbolisches Modell zur Darstellung mechanischer Zusammenhänge vor.

Generische Roboteraktionen sind durch ihre Trajektorien definiert. Um diese im symbolischen Model behandeln zu können, wird eine Abbildung auf Aktionssymbole mit Vor- und Nachbedingungen entwickelt.

Symbolisierung Es werden Verfahren entwickelt, die Symbole des symbolischen Modells aus dem kontinuierlichen Szenenmodel zu erzeugen.

Ausführungsüberwachung Es wird eine Methode entwickelt, um den Erfolg einer Ausführung in Echtzeit zu überwachen. Mittels eines überwachten Lernverfahrens wird ein Klassifikator auf Sensordaten erstellt, der in ein Echtzeit Monitoring Framework integriert wird.

Demonstrator Ein Manipulationsdemonstrator wird im Rahmen der Arbeit zu einem zweihändigen System erweitert. Dies beinhaltet insbesondere den Ausbau der Steuer-

ungssoftware, die dazu um Module für die entwickelten Methoden erweitert wird. Der Demonstrator wird verwendet, um die Ergebnisse der Arbeit zu evaluieren.

1.3. Aufbau der Arbeit

Im folgenden Kap. 2 werden die Grundlagen dieser Arbeit beschrieben, also Verfahren auf denen die Arbeit aufbaut und die von ihr verwendet werden. Im Gegensatz dazu beschreibt Kap. 3 den Stand der Technik, Verfahren aus Bereichen, zu denen diese Arbeit einen Beitrag liefert und mit denen sie sich vergleicht.

In Kap. 4 werden die Anforderungen an eine Lösung der beschriebenen Problemstellung analysiert. Es entsteht ein formales Modell von Planung in symbolischen und kontinuierlichen Modellen. Aus dem Modell werden Anforderungen an das System abgeleitet, darauf basierend wird ein Konzept zur Adaption szenenabhängiger Manipulationssequenzen entwickelt. Komponenten des Konzeptes sind in den folgenden Kapiteln beschrieben. Kap. 5 beschreibt das symbolische Modell und die Planung darauf. In Kap. 6 wird die autonome Herleitung der Symbole des symbolischen Modells aus der kontinuierlichen Darstellung vorgestellt. Kap. 7 beschäftigt sich mit der Abbildung der Symbole auf die Ausführung und der Überwachung der Konsistenz der Ausführung mit dem geplanten Handlungsverlauf.

In Kap. 8 werden die Ergebnisse dieser Arbeit experimentell evaluiert. Die Arbeit endet mit einer Zusammenfassung in Kap. 9.

2. Grundlagen

In diesem Kapitel werden die Grundlagen dieser Arbeit beschrieben. Es beschränkt sich auf Bereiche, zu denen die Arbeit keinen eigenen Beitrag liefert.

Diese Arbeit verwendet Planungsverfahren aus unterschiedlichen Planungsdomänen, um Problemstellungen auf unterschiedlichen Abstraktionsebenen zu lösen. Dieses Kapitel erhebt nicht den Anspruch, den Stand der Technik in den beschriebenen Gebieten vollständig aufzuzeigen. Stattdessen liegt der Fokus darauf, die Eigenschaften verwendeter Verfahren zu beleuchten. Weiter werden verbreitete Verfahren vorgestellt, um im Vergleich die Vorteile der verwendeten Verfahren herauszuarbeiten.

Im Abschnitt 2.1 werden Domänenmodelle beschrieben, die zur symbolischen Beschreibung von Handlungen und Weltzuständen verwendet werden. Darauf aufbauend werden in Abschnitt 2.2 Verfahren beschrieben, die Handlungssequenzen aus symbolischen Modellen erzeugen. Analog werden im Anschluss in Abschnitt 2.3 Modelle im Kontinuierlichen beschrieben sowie darauf aufbauend, in Abschnitt 2.4, Planungsverfahren im Kontinuierlichen. In Abschnitt 2.5 werden Eigenschaften von Physiksimulationen skizziert. Das Kapitel endet mit einem Fazit in Abschnitt 2.8.

2.1. Domänenmodelle im Symbolischen

2.1.1. STRIPS und ADL

Eine Beschreibung für symbolische Weltzustände und Operatoren auf diesen führen (Fikes u. Nilsson, 1971) mit der *STRIPS*[1] Repräsentation ein. Eine wesentliche Feststellung ist, dass ein Operator beim Überführen von W_s^n nach W_s^{n+1} meist nur wenige Eigenschaften des Zustandes verändert. Das Erzeugen eines unveränderten Folgezustandes ist ein Problem, wenn die Zustandsbeschreibung auf Prädikatenlogik basiert, es wird dann als Frame Problem bezeichnet (McCarthy u. Hayes, 1969). STRIPS nutzt daher keinen rein Prädikatenlogik-basierten Ansatz.

[1] STanford Research Institute Problem Solver

Stattdessen wird ein Zustand in STRIPS als Liste von erfüllten Prädikaten modelliert, z. B.:

$$
\begin{aligned}
W_s = & \\
(\quad & P(x,y), \\
& Q(y), \\
& \vdots \\
) &
\end{aligned}
$$

Dabei sind nur positive, konjunktiv verknüpfte Prädikate erlaubt. Es gilt die Annahme einer geschlossenen Welt[2], d. h. nicht aufgeführte Prädikate sind falsch. Dies bedeutet insbesondere, dass das Nicht-Wissen einer Eigenschaft nicht modelliert werden kann, der Wahrheitswert jedes Prädikats, das verwendet werden soll, muss bekannt sein.

Um Operatoren zu ermöglichen, welche die Attribute der Welt implizit unverändert lassen, wird ein Operator wie folgt aus drei Teilen definiert:

Vorbedingung[3] beschreiben die Prädikate, die in einem Zustand erfüllt sein müssen, damit die Aktion ausgeführt werden kann.

Add-Liste Prädikate, die dem Folgezustand durch das Ausführen der Aktion hinzugefügt werden.

Delete-Liste Prädikate, die aus dem Folgezustand durch das Ausführen der Aktion entfernt werden.

Dabei gilt die STRIPS-Annahme. Sie besagt, dass Prädikate, die in der Definition des Operators nicht erwähnt werden, unverändert bleiben. Formal wird ein Operator wie folgt dargestellt:

$$
\begin{aligned}
\text{Action} & \\
(\quad & \text{Name}(x,y), \\
\text{PRECOND} \ : \ & P(x,y) \wedge \ldots \\
\text{ADD} \ : \ & Q(y), \ldots \\
\text{DELETE} \ : \ & R(x), \ldots \\
) &
\end{aligned}
$$

Auch hier ist nur die konjunktive Verknüpfung erlaubt.

Die Einschränkungen von STRIPS werden in darauf aufbauenden Darstellungen reduziert. Beispielhaft sei hier die *Action Description Language* (ADL)(Pednault, 1986) aufgeführt. Ein Vergleich der beiden Sprachen ist in Tabelle 2.1 dargestellt.

[2] engl.: closed-world assumption

STRIPS Sprache	**ADL Sprache**
Nur positive Literale in den Zuständen	Positive und negative Literale in den Zuständen
Geschlossene Welt Nicht aufgeführte Literale sind falsch	Offene Welt Nicht aufgeführte Literale sind unbekannt
P in Add-List $\Longrightarrow$ P zum Zustand hinzufügen	P in Add-List $\Longrightarrow$ zum Zustand hinzufügen und $\neg$P löschen
P in Delete-List $\Longrightarrow$ P aus Zustand löschen	P in Delete-List $\Longrightarrow$ aus Zustand löschen und P hinzufügen
Nur Konstanten als Ziel	Quantifizierte Variablen : $\exists x : \ldots$
Nur Konjunktionen als Ziel	Konjunktion, Disjunktion und Negation
Keine bedingten Effekte	Bedingte Effekte: Falls P dann E
Keine Unterstützung für Gleichheit	Eingebautes Äquivalenzprädikat $x = y$
Keine Unterstützung für Typen	Typisierte Variablen, x : Block

Tab. 2.1.: Gegenüberstellung der Sprachen STRIPS und ADL nach (Russell u. Norvig, 2003).

2.1.2. Hierarchische Aufgaben Netzwerke

Hierarchische Aufgaben Netzwerke[4], (HTN) modellieren Handlungssequenzen. Sie bieten keinen Mechanismus um Weltzustände abzubilden. Sie sind eine Darstellung, um komplexe abstrakte Aufgaben effizient in Sequenzen von abstrakten Teilaufgaben zu zerlegen, die in ihrer Summe das Ziel einer Gesamtaufgabe erreicht.

HTN vermeiden es, für eine Probleminstanz einen Graph des Handlungsraums zu erstellen und zu durchsuchen. Stattdessen modellieren sie bekannte Lösungswege für häufig wiederkehrende (Teil-)Probleme. Ein Aufgabe wird dabei in einfachere Teilaufgaben zerlegt. Die Zerlegung führt zu einem Graphen in dem Aufgaben als Knoten repräsentiert sind, dem HTN. Die übergeordneten Aufgaben sind mit ihren Teilaufgaben durch gerichtete Kanten verbunden. Teilaufgaben einer Gesamtaufgabe können in unterschiedlichen Beziehungen stehen. Üblich sind Sequenzen, d. h. die Teilaufgaben werden nacheinander ausgeführt. Weiter können Alternativen modelliert werden. Zur Auswahl von Alternativen muss auch das HTN eine Anbindung an den Weltzustand haben, etwa über Vorbedingungen einzelner Teilaufgaben, wie in der STRIPS Darstellung. Umsetzungen von HTN können auch weitere Beziehungen enthalten, wie etwa parallel ausführbare Aktionen.

Ein HTN für die Aufgabe des Transports eines Objektes durch einen Manipulator ist in Abb. 2.1 dargestellt. Hierarchische Aufgabennetze stellen die Grundlage von HTN Plannern wie in (Nau u. a., 2003) beschrieben dar. Weitere Umsetzungen mit Robotik-Anwendung sind die Task Description Language (Simmons, 1994; Simmons u. Apfelbaum, 1998) und darauf aufbauend die

[4] engl.: Hierarchical Task Networks

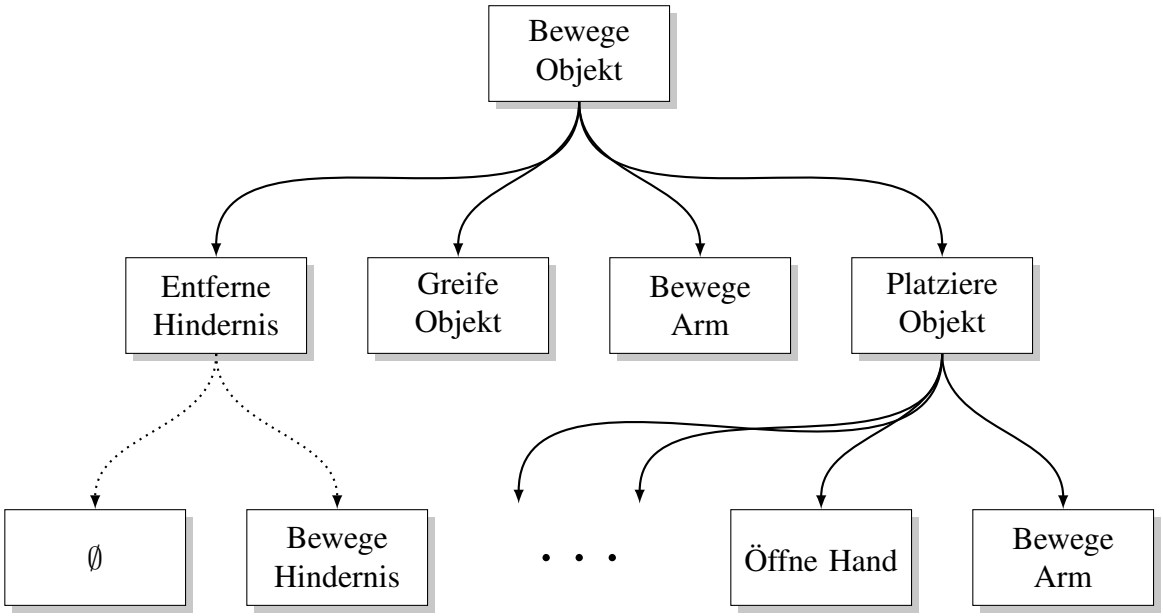


Abb. 2.1.: Visuelle Darstellung eines hierarchischen Aufgabennetzes. Die Aufgabe wird in mehrere Teilaufgaben zerlegt. Durchgehende Linien verbinden Kindknoten, die Sequenzen modellieren. Gepunktete Linien verbinden Alternativen. Im Beispiel kann die Teilaufgabe „Entferne Hindernis" durch eine leere Aktion ausgeführt werden, wenn kein Hindernis vorhanden ist.

Flexiblen Programme (Knoop, 2007; Knoop u. a., 2006), die im folgenden Abschnitt genauer beschrieben werden.

Um HTN-basierte Planung einzusetzen, müssen aufgabenspezifische Dekompositionen von einem Experten erstellt werden. Weiter eignen sie sich schlecht zur automatischen Berücksichtigung von Parallelität, wie sie zum Verteilen von Aufgaben auf mehrere Manipulatoren benötigt wird. Daher entscheidet sich diese Arbeit gegen den Einsatz von HTN.

2.1.3. Flexible Programme

Flexible Programme (FP), wie sie von (Knoop u. a., 2006; Knoop, 2007) eingeführt werden, stellen eine Repräsentation von Aufgabenwissen für das Handlungslernen von Servicerobotern dar. Sie basieren auf den im vorigen Abschnitt vorgestellten Hierarchischen Aufgaben Netzwerken. Wie bei HTN spezifiziert ein Baum die Ausführungswege einer Aufgabe, einzelne Knoten entsprechen Teilaufgaben. Blätter eines FP stellen *atomare Handlungen* dar. Sie werden auch Elementaroperatoren genannt. Ein innerer Knoten des FP wird als 7-Tupel spezifiziert:

$$\mathcal{P} = \left\{ \mathrm{Id}, \mathrm{C_{pre}}, \mathrm{C_{post}}, \mathrm{C_{rt}}, \mathrm{R}, \mathrm{S}, \mathrm{P} \right\} \tag{2.1}$$

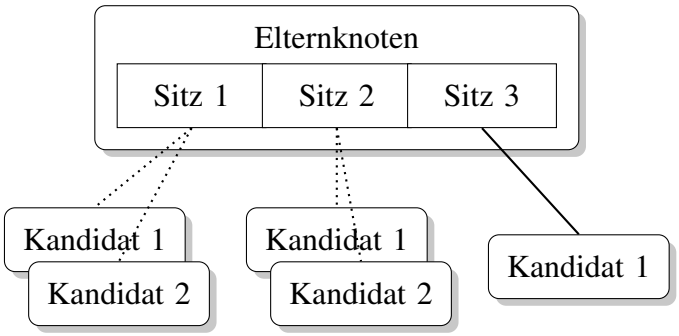


Abb. 2.2.: Schematische Darstellung eines Flexiblen Programms (FP). Die Folge von Sitzen im Elternknoten modelliert eine Sequenz. Stehen für einen Sitz mehrere Kandidaten zur Auswahl, ist dadurch eine Alternative modelliert. Nach (Knoop, 2007)

Dabei bezeichnet Id einen eindeutigen Namen des (Teil–) FP, C_{pre} die *Vorbedingungen*[5], die erfüllt sein müssen, damit der Knoten ausgeführt werden kann. C_{post} beschreibt *Nachbedingungen*[6], mit denen nach der Ausführung der Erfolg überprüft werden kann. C_{rt} beschreibt die *Laufzeitbedingungen*[7], die während der Ausführung erfüllt sein müssen. R[8] gibt eine *Bewertungsfunktion* an, die verwendet werden kann, um einen Knoten aus einer Menge möglicher, ausführbarer Knoten zu selektieren. Analog gibt S (Success) ein Bewertungsmaß für den Erfolg der Ausführung an. P stellt den sogenannten *Prospekt* des Knoten dar, der seine Kindknoten und deren Zusammenhänge untereinander darstellt (Abb. 2.2). Dazu enthält der Prospekt eine Liste von *Sitzen*, die der Reihe nach zur Ausführung kommen. Jeder Sitz enthält eine nicht leere Menge von *Kandidaten*, von denen pro Sitz genau einer zur Ausführung kommt. Schließlich können Sitze zu Gruppen zusammengefasst werden. Sitze in einer Gruppe werden parallel ausgeführt.

Die Abarbeitung eines Flexiblen Programms erfolgt entlang einer Tiefensuche. Der Gesamtbaum spannt jede mögliche Ausführung auf. Beim Betreten eines Knotens während der Tiefensuche entscheidet die Ausführung mittels der oben beschrieben Mechanismen, welche Elemente des Prospekts zur Ausführung kommen. Zur symbolischen Modellierung steht dem System dabei eine Datenbank aus Schlüssel-Werte-Paaren zur Verfügung, in der mittels Prädikatenlogik erster Ordnung Bedingungen formuliert werden können.

2.1.4. Planning Domain Definition Language

Die *Planning Domain Definition Language* (PDDL), (McDermott u. a., 1998) ist eine Sprache zur Beschreibung von Planungsdomänen, die beim Internationalen Planungswettberwerb (IPC, 2014) eingesetzt wird und daher bei vielen symbolischen Planern zum Einsatz kommt.

[5] engl.: preconditions
[6] engl.: postconditions
[7] run-time conditions
[8] engl.: rating

PDDL unterstützt zu STRIPS ähnliche Operatoren, konditionale Effekte, Quantoren sowie die hierarchische Zerlegung von Aktionen. Dies bietet konzeptionell keinen Mehrwert zu den bereits vorgestellten Konzepten. Der Nutzen von PDDL findet sich stattdessen in der klar spezifizierten, verbreiteten syntaktischen Struktur der Sprache. Sie ermöglicht es, unterschiedliche Planer-Implementierungen auf ein Problem anzuwenden.

Eine Aktion in PDDL kann aussehen wie im folgenden Beispiel aus (McDermott u. a., 1998). Die Aktion „put-in" stellt das Objekt x an den Ort l.

```
(:action put-in
    :parameters (?x - physob ?l - location)
    :precondition (not (= ?x B))
    :effect (when (and (at ?x ?l) (at B ?l)) (in ?x)) )
```

2.1.5. New Domain Definition Language

Die *New Domain Definition Language* (NDDL) (Frank u. Jonsson, 2002; Daley u. a., 2005; Bernardini u. Smith, 2007) ist eine von der NASA entwickelte Domänenbeschreibung. Sie verzichtet auf die aus STRIPS bekannte Modellierung von Zuständen und Aktionen. Stattdessen werden *Token* und *Zeitstrahlen*[9] verwendet. Außerdem wird eine Zeitdiskretisierung eingeführt, bei der Zeitpunkte mit Ganzzahlen dargestellt werden. Ein Token bezeichnet ein Prädikat zusammen mit einem Anfangs- und einem Endzeitpunkt. Zeitstrahlen verbinden Token zu Zustandsvariablen. Ein Zeitstrahl vereinigt eine Menge von Prädikaten in der Art, dass zu jedem Zeitpunkt genau ein Prädikat der Menge gültig sein muss. Auf diese Weise lassen sich Zeitstrahlen zum Modellieren von Eigenschaften verwenden, denen immer genau ein Wert zugewiesen sein muss. Ein Beispiel dafür ist etwa der Ort eines Objektes. Gleichzeitig kann ein Zeitstrahl auch als eine Ressource betrachte werden, die in der Lage ist, Aufgaben auszuführen. So kann ein Zeitstrahl „Manipulator" z.B. durch das Prädikat „Greifen" oder eine andere Aktion belegt sein, oder er befindet sich in einem Ruhezustand. Es kann ein gemischter Zeitstrahl verwendet werden. z.B. kann ein „Ort" Zeitstrahl eines Objektes sowohl den Zustand „An Ort", der mit einem Ort verknüpft ist, als auch den „In Hand" Zustand haben, der in einer Passiv-Konstruktion modelliert, dass das Objekt gerade manipuliert wird.

Um Beziehungen zwischen Zeitstrahlen zu modellieren werden *Allen Relationen* verwendet (Allen, 1983). Allen Relationen beschreiben einen zeitlichen Zusammenhang von Token aufgrund von Relationen zwischen deren Anfangs- und Endzeitpunkt. So kann etwa modelliert werden, dass eine Aktion vor einer anderen, oder auch genau gleichzeitig stattfinden muss. Eine Auswahl an Allen Relationen, die von NDDL unterstützt wird, ist in Tabelle 2.2 aufge-

[9] engl.: timeline

Bezeichnung dt.	Bezeichnug engl.	Beziehung		
vor	before	Ursprung.Ende	$\leqslant$	Ziel.Anfang
nach	after	Ziel.Ende	$\leqslant$	Ursprung.Anfang
trifft	meets	Ursprung.Ende	$=$	Ziel.Anfang
getroffen von	met_by	Ursprung.Anfang	$=$	Ziel.Ende
gleich	equal or equals	Ursprung.Anfang	$=$	Ziel.Anfang $\wedge$
		Ursprung.Ende	$=$	Ziel.Ende $\wedge$
		Ursprung.Dauer	$=$	Ziel.Dauer
enthält	contains	Ursprung.Anfang	$\leqslant$	Ziel.Anfang $\wedge$
		Ziel.Ende	$\leqslant$	Ursprung.Ende $\wedge$
		Ursprung.Dauer	$\geqslant$	Ziel.Dauer
enthalten von	contained_by	Ziel.Anfang	$\leqslant$	Ursprung.Anfang $\wedge$
		Ursprung.Ende	$\leqslant$	Ziel.Ende $\wedge$
		Ursprung.Dauer	$\leqslant$	Ziel.Dauer
beginnt während	starts_during	Ursprung.Anfang	$\geqslant$	Ziel.Anfang $\wedge$
		Ursprung.Anfang	$<$	Ziel.Ende
beliebig	any		keine	

Tab. 2.2.: Auswahl einiger Allen-Relationen(Allen, 1983), die von NDDL unterstüzt werden.

führt. Damit können Aktionen auf Zeiträume und Ressourcen abgebildet werden. NDDL eignet sich damit besonders für sog. Scheduling Probleme.

2.2. Planung im Symbolischen

Im symbolisch beschriebenen Zustandsraum kann aus einem Zustand durch Anwenden der Aktions-Operatoren eine Menge von Folgezuständen erzeugt werden. Analog kann eine Menge von Vorgängerzuständen erzeugt werden. Durch rekursives Anwenden dieses Verfahrens kann ausgehend von einem Startzustand (analog: Zielzustand) ein Zustandsbaum aufgebaut werden. Knoten des Baums sind die Zustände, Kanten die Aktionen. Der Baum enthält alle durch Aktionen erreichbare Zustände. Die Planung einer Aktionssequnenz reduziert sich auf die Suche nach einem Pfad, der in diesem Baum vom Start– zum Zielzustand führt. Zur Suche im Baum können bekannte Suchalgorithmen, wie Tiefen– oder Breitensuche, verwendet werden, ebenso heuristikbasierte Verfahren, wie die A*-Suche.

In der Praxis ist es im Allgemeinen nicht möglich, den Zustandsbaum vollständig aufzubauen, da dieser, abhängig vom Zustandsraum, unendlich groß wird. Um den Suchraum und damit den Aufwand der Planung zu reduzieren, sind eine Vielzahl von Verfahren bekannt (Russell u. Norvig, 2003; Hertzberg u. Chatila, 2008). Im Folgenden wird eine Auswahl für diese Arbeit relevanter Verfahren beschrieben.

2.2.1. Lineare Planer und Planung auf partiell geordneten Operatoren

Ein wesentlicher Faktor, der für die Größe des Suchraums bei symbolischen Planungsproblemen verantwortlich ist, ist die Reihenfolge der Aktionen. Angenommen es gibt eine Menge mit n unabhängigen Aktionen, die alle im Plan vorkommen können. Dies kann z.B. der Fall sein, wenn n Objekte von einem Tisch auf einen anderen geräumt werden sollen. Dann gibt es $n!$ mögliche Reihenfolgen diese auszuführen. Für das Ergebnis des Plans ist die Reihenfolge jedoch irrelevant. Um den Suchbaum zu reduzieren, sind zwei Ansätze verbreitet. Zum einen kann eine Reihenfolge der Aktionen erzwungen werden, dieser Ansatz führt zur linearen Planung. Oder es wird eine Modellierung verwendet, die es erlaubt, die Unabhängigkeit von Aktionen darzustellen und es dementsprechend vermeiden kann, eine Reihenfolge dieser Aktionen zu planen. Dies ist als partiell geordnetes Planen bekannt.

Der bekannteste lineare Planer ist der STRIPS Planer (Fikes u. Nilsson, 1971), dessen Domänenbeschreibung bereits in Abschnitt 2.1.1 vorgestellt wurde. STRIPS arbeitet die Ziele des Plans in der Reihenfolge ab, in der diese definiert sind. Sind die Ziele bzw. Aktionen voneinander abhängig, so spricht man von *Interaktion*. Es wird zwischen positiver und negativer Interaktion zweier Aktionen unterschieden. Bei positiver Interaktion erfüllt eine Aktion auch das Ziel einer anderen, bei negativer zerstört eine Aktion ein bereits erreichtes Teilziel. Interaktion wird bei STRIPS nur in dem Sinne berücksichtigt, dass durch vorige Aktionen bereits erfüllte Teilziele nicht weiter behandelt werden müssen. Das Ignorieren von Interaktion führt zu den Hauptnachteilen des STRIP Planers. So existieren simple Planungsprobleme, die durch STRIPS nicht gelöst werden können, obwohl einfache Lösungen existieren. Durch negative Interaktion können suboptimale Lösungen erzeugt werden, besonders zu erwähnen ist hier die Sussmann Anomalie (Sussman, 1973; Russell u. Norvig, 2003)

Partial Order Planner (POP), (Weld, 1994) basieren auf einer partiell geordneten Darstellung eines Plans. Ein partiell geordneter Plan ist in Abb. 2.3 veranschaulicht. Formell besteht er aus einer Menge von Aktionen und einer Menge Ordnungsrelationen $A \prec B$. In Abb. 2.3 sind die Ordnungsrelationen durch gerichtete Kanten dargestellt. Die Relationen werden mit den Eigenschaften, die durch sie bereit gestellt werden, annotiert. Die Planung erfolgt durch nichtdeterministisches Hinzufügen von Aktionen zum Plan, die offene Bedingungen erfüllen. Für hinzugefügte Aktionen werden Konflikte durch Hinzufügen von Ordnungsrelationen aufgelöst. Für den genauen Verlauf des Verfahrens sei auf (Russell u. Norvig, 2003) verwiesen. Vorteilhaft bei POP ist neben dem reduzierten Aufwand, dass aus dem erstellten Plan Informationen über Nebenläufigkeit von Aktionen gewonnen werden. Allerdings berücksichtigt das Verfahren keine Allokation von Ressourcen, die Aktionen parallel ausführen können.

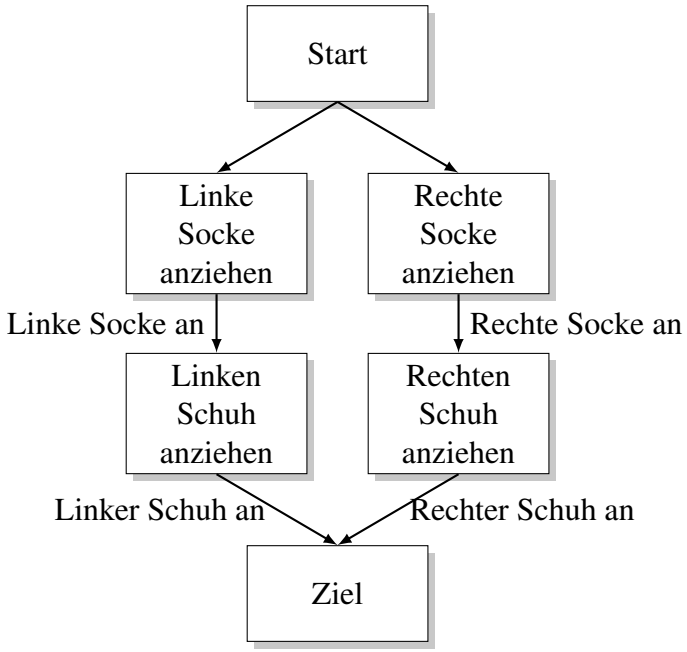


Abb. 2.3.: Beispiel für eine partiell geordnete Darstellung eines Plans aus (Russell u. Norvig, 2003)

2.2.2. Hierarchische Planer

HTN Planer generieren aus einer Menge von Operatoren eine Sequenz von Aktionen, die eine Aufgabe erfüllt. Dabei ist der wesentlicher Teil der Planung nicht im Planer, sondern in den Operatoren spezifiziert (vgl. Abschnitt 2.1.2). Diese werden rekursiv durch weitere Operatoren aufgelöst, sodass eine Sequenz von atomaren Aktionen entsteht. Ein Algorithmus, der diese Auflösung durchführt, ist in Algorithmus 2.1 skizziert. Der einfache Algorithmus bringt jedoch ähnliche Nachteile mit sich, wie die lineare Planung: Die Reihenfolge unabhängiger Teilziele ist in Operatoren fest vorgegeben, ebenso wird die Interaktion zwischen unterschiedlichen Aktionen nicht berücksichtigt.

Das HTN Planungssystem SHOP 2 (Nau u. a., 2003) verwendet daher Methoden des Planens mit partiell geordneten Operatoren. Statt einer Liste von Aktionen wird eine Menge von Aktionen und Abhängigkeiten generiert. Konflikte werden wie bei POP aufgelöst, ebenso können Aktionen verschmolzen werden, um positive Interaktion zu erlauben.

2.2.3. Refinement Search

Die *Constraint Satisfaction-basierte Planung* verwendet ein Zeitstrahl bzw. *Token* basiertes Modell. Es wurde in Abschnitt 2.1.5 vorgestellt. Die Planung erfolgt, indem zuerst ein unvollständiger Plan erstellt wird, der alle Tokens des Start– und Zielzustands sowie alle Objekte enthält. Alle Tokens sind aktiv. In jedem Schritt werden nun für alle Bedingungen der aktiven

Algorithmus 2.1 Einfacher Algorithmus zur Dekomposition von HTNs

```
1:  function HTN PLANER(Operator, W_s)
2:      if ¬Anwendbar(Operator, W_s ) then
3:          return (False, False)
4:      else if Operator ist atomar then
5:          return ((Operator), Operator(W_s))
6:      else if Operator hat keine Sitze then
7:          return ((), W_s)
8:      else
9:          Plan ← ()
10:         for Sitz ∈ (Operator → Sitze) do
11:             Done ← false
12:             while ∃A ∈ Sitz : A ist unbearbeitet ∧¬ Done do
13:                 Wähle unbearbeitete Auswahl aus Sitz→ A
14:                 (P′, W_s′) ← HTN Planer(A, W_s )
15:                 if P′ ≠ False then
16:                     Plan ← (Plan, P′)
17:                     W_s ← W_s′ )
18:                     Done ← True
19:                 end if
20:             end while
21:             if ¬ Done then
22:                 return (False,False)
23:             end if
24:         end for
25:         return (Plan, W_s)
26:     end if
27: end function
```

Token, inaktive, sog. Slave-Tokens erzeugt. Dann wird nach *Makeln*[10] im Plan gesucht. Dies sind :

Ungebundene Variablen Dem Domänenmodell liegt ein Constraint Satisfaction Modell zugrunde. Dort können Variablen einen Wertebereich statt eines einzelnen Wertes haben. In einem vollständigen Plan ist das nicht erlaubt. Der Makel kann behoben werden, indem ein möglicher Wert ausgewählt wird.

Offene Bedingungen Offene Bedingungen sind vorhanden, solange inaktive Token im unvollständigen Plan vorhanden sind. Um den Makel zu beheben, können Token aktiviert oder mit anderen, aktiven und kompatiblen Token verschmolzen werden.

Verlaufs-Makel Tokens können über Objekte Anforderungen aneinander haben. Diese werden durch neue Ordnungsbedingungen aufgelöst.

[10] engl.: flaw

Dieser Schritt wird wiederholt, bis der Plan vollständig ist. Dabei kann der Planer in Sackgassen laufen. In diesem Fall wird ein sukzessives Backtracking durchgeführt. Eine detaillierte Beschreibung des Verfahrens findet sich in (Bedrax-Weiss u. a., 2004).

Planung durch Refinement Search empfiehlt sich vor allem durch ihre Fähigkeit zum Schedulen von Aktionen, d. h. für Aktionen können Ausführungszeiträume bestimmt werden und sie können auf Ressourcen verteilt werden. Werden diese Fähigkeiten nicht benötigt, so entsteht durch die Modellierung und Planung der zeitlichen Zusammenhänge ein Zuwachs des Rechenaufwands.

Aufgrund der Anforderung dieser Arbeit, Aufgaben auf ausführende Ressourcen zu verteilen, wird der Ansatz weiter in Betracht gezogen. Die *Extensible Universal Remote Operations Planning Architecture for Planning, Scheduling, Constraint Programming and Optimization* (EUROPA PSO) (Daley u. a., 2005) bietet eine Implementierung.

2.2.4. Probabilistische Planung

Für Handlungsadaption ist eine einzelne Entscheidung nicht ausreichend, es wird nach einer Sequenz von Aktionen gesucht. Dementsprechend benötigt man eine Sequenz von Entscheidungen für Aktionen. Planung von Aktionen mit probabilistischen Effekten kann als *Markov-Enscheidungsprozess*[11] (MDP) modelliert werden. Dieser besteht neben einem initialen Zustand W_s^0 und einer Menge von möglichen Aktionen A aus einem *Transitionsmodell* T und einer *Belohnungsfunktion*[12] B:

$$\begin{aligned} T(W_s^t, a, W_s^{t+1}) &\rightarrow [0,1] \text{ mit } a \in A \\ B(W_s^t) &\rightarrow \mathbb{R} \end{aligned} \tag{2.2}$$

Das Transitionsmodell beschreibt dabei die Wahrscheinlichkeit, dass ein Zustand W_s^t in einen Zustand W_s^{t+1} übergeht. Dabei liegt die *Markov-Annahme* zugrunde. Sie besagt, dass ein Zustand nur von einer endlichen Anzahl an Vorgängerzuständen abhängt. Da in T nur ein Vorgängerzustand modelliert ist, handelt es sich um einen Markov Prozess erster Ordnung. Die Bewertungsfunktion B beschreibt eine Belohnung oder Bestrafung für das Erreichen eines Zustands. Ziel der Planung ist es, eine Aktionssequenz zu generieren, welche die Belohnung maximiert. Dazu werden Regeln[13] $\pi(s)$ erzeugt, die für jeden Zustand eine auszuführende Aktion definiert. Solche Regeln können beispielsweise durch den „value itteration" Algorithmus gefunden werden (Russell u. Norvig, 2003).

[11] engl.: Markov Decision Process
[12] engl.: reward function
[13] engl.: policy

Der Markov-Entscheidungsprozess nimmt einen bekannten Weltzustand W_s an. Für die Handlungsadaption kann man weiter gehen und auch noch den Zustand, in dem sich die Welt befindet, als unsicher betrachten. Diese Situation kann durch einen teilweise beobachtbaren Markov-Entscheidungsprozess *POMDP*) modelliert werden. POMDPs besitzen die selben Elemente wie MDPs. Zusätzlich haben sie ein *Beobachtungsmodell* [14] O:

$$O(W_s^t, o) \rightarrow [0,1] \tag{2.3}$$

Es beschreibt die Wahrscheinlichkeit, dass eine Beobachtung o im Zustand W_s^t wahrgenommen wird. Ein POMDP kann auf ein MDPs abgebildet werden, indem der „believe state" des Planers als Weltzustand verwendet wird. Dieser ist per Definition bekannt. Allerdings handelt es sich dabei um einen kontinuierlichen Raum, sodass die Methoden der MDP nicht einfach übernommen werden können. Optimale Entscheidungsregeln für ein POMDP zu finden ist PSPACE hart (Russell u. Norvig, 2003) und damit schon für kleine Zustandsräume nicht mehr möglich. Aufgrund der online Anforderung werden POMDP für diese Arbeit nicht weiter in Betracht gezogen.

2.3. Domänenmodelle im Kontinuierlichen

2.3.1. Szenenmodelle

Szenenmodelle beschreiben die Umgebung, in der ein Roboter agiert. Grundsätzlich kann in einem Szenenmodell zwischen zwei Arten von Entitäten unterschieden werden. Entitäten, die durch den Roboter beeinflussbar sind und zu denen Hintergrundwissen existiert, werden als Objekte bezeichnet. Dagegen werden nicht beeinflussbare Entitäten als Hindernisse bezeichnet.

Hindernisse generieren im Modell lediglich das Wissen ihrer materiellen Existenz. Es ist kein Zusammenhang innerhalb der Entität oder zwischen verschiedenen Entitäten bekannt. Zwei unterschiedliche Gründe können eine Entität zum Hindernis machen: Entweder, es fehlt das Hintergrundwissen, um das Hindernis zum Objekt zu machen, z.B. bei aus Punktwolken wahrgenommenen unbekannten Entitäten. Oder der Roboter könnte das Objekt aufgrund seiner Eigenschaften nicht beeinflussen, sodass es nicht sinnvoll ist, zugehöriges Objektwissen zu modellieren. Beispielsweise wird eine Wand, auch wenn Objektwissen modellierbar ist, als Hindernis betrachtet. Aufgrund des fehlenden Hintergrundwissens über Hindernisse ist es nicht sinnvoll, diese in die symbolische Ebene zu abstrahieren. Ihre Berücksichtigung erfolgt stattdessen auf der kontinuierlichen Ebene. Auf der anderen Seite ist das Generieren von Hindernismodellen

[14] engl.: observation model

aus geeigneten Sensordaten (z. B. 3D Punktwolken) sehr einfach, da keine Interpretation durchgeführt werden muss.

Hindernisse besitzen nur für die Kollisionsvermeidung Relevanz. Entsprechend modelliert das Hindernismodell, welcher Bereich der Szene von Hindernissen belegt ist. Dies kann z. B. mit Dreiecksnetzen, Voxel-basierten Verfahren oder umschließenden geometrischen Primitiven geschehen. Details zur Hindernismodellierung finden sich in der Literatur zur kollisionsfreien Bahnplanung, etwa in (LaValle, 2006).

Im Bereich der Objektmodellierung kennt der Stand der Technik eine Vielzahl von Modellen, die meist aufgabenspezifisch motiviert sind. Ein vollständiger Überblick würde den Rahmen dieses Abschnitts sprengen, es sei hier auf (Becher, 2008; Kasper, 2013) verwiesen.

Eine Strukturierung von modellierten Objekteigenschaften wird in (Bogoni u. Bajcsy, 1995) vorgeschlagen:

Geometrisches Modell Analog zu Hindernismodellen kann auch für Objekte ein geometrisches Modell verwendet werden. Neben der Kollisionsvermeidung kommt dieses auch bei der Greifplanung zum Einsatz. Eine weitere Verwendung ist die Objekterkennung, meist in Kombination mit Materialeigenschaften. Zur Manipulation ist die Objektposition in der Szene eine der wichtigsten Informationen. Sie ergänzt Objektmodelle und Hindernismodelle zum Szenenmodell.

Materialeigenschaften Sie beschreiben Objekteigenschaften, die aus dem Material des Objektes resultieren. Dazu zählen vor allem visuelle Eigenschaften der Oberfläche wie Farbe, Reflektanz und Textur. Weiter fallen in diese Kategorie auch Eigenschaften, die für die Physik eines Objektes relevant sind, insbesondere Reibungskoeffizient und Dichte des Materials.

Kinematisches Modell Es beschreibt Gelenke und Zwangsbedingungen der Bewegung eines Objektes. In dieser Arbeit wird nur für den Manipulator ein kinematisches Modell aufgestellt; Objekte werden als Starrkörper angenommen.

Dynamisches Modell Es enthält Informationen über Kräfte, Momente, Impulse sowie Masse und Trägheitstensor eines Objektes. Es ist für die Physik des Objektes relevant.

Objekt- und Hindernismodell ergeben zusammen das Szenenmodell. Dieses beinhaltet neben den individuellen Eigenschaften der Objekte auch noch Beziehungen zwischen diesen. Da diese Arbeit einen Beitrag zur Modellierung von mechanischen Beziehungen zwischen Objekten liefert, wird der Stand der Technik auf diesem Gebiet ausführlich in Abschnitt 3.1.2 dargestellt.

2.3.2. Arbeitsraumbeschreibung

Zur Planung von Manipulation benötigt man ein Modell, das eine Beziehung zwischen dem kartesischen Raum und dem Konfigurationsraum eines Manipulators herstellt. Einen direkten Bezug zwischen einzelnen Punkten der Räume bieten die Methoden der direkten und inversen Kinematik. Diese beschreiben, ob ein Punkt im Arbeitsraum vom Manipulator erreicht werden kann. Sie sind jedoch ungeeignet, um Aussagen über ganze Bereiche des Raumes zu treffen. Betrachtet werden sollen hier Methoden, die eine Aussage in der Form ermöglichen: Hat ein Punkt oder ein Bereich eine positive Erwartung bezüglich seiner Erreichbarkeit, so gibt es eine bekannte Menge benachbarter Punkte, für die ebenfalls eine positive Erreichbarkeit zu erwarten ist. Analoge sollte das für eine negative Aussage über die Erreichbarkeit gelten.

Eine konzeptuell einfache Möglichkeit eine solche Beschreibung zu realisieren ist die konvexe Hülle des Arbeitsraums. Sie erlaubt die klare Aussage, dass alle Punkte außerhalb der Hülle nicht erreichbar sind. Innerhalb der Hülle werden alle Punkte als erreichbar angenommen, was jedoch die Kinematik des Roboters nicht korrekt darstellt. Eine solche Darstellung eignet sich für einen Vorverarbeitungsschritt für die inverse Kinematik, um die möglicherweise aufwändige Berechnung für Punkte zu ersparen, die für den Arm unmöglich erreichbar sein können.

Ein numerisches Maß, das beschreibt, wie gut ein Manipulator an einem gegebenen Punkt manipulieren kann, also wie gut er seinen Tool Center Point (TCP) im Raum plazieren kann, ist der *Manipulierbarkeits- Ellipsoid*[15]. Dabei wird ausgenutzt, dass der Rang R der Jakobimatrix J_T eines Manipulators außerhalb einer Singularität gleich ihrer Dimension M ist. In einer Singularität dagegen gilt $R < M$ und damit $|J_T| = 0$. Basierend auf den Eigenvektoren der Singulärwertzerlegung, skaliert mit den Singulärwerten der Jakobimatrix, definiert (Yoshikawa, 1985) den Manipulierbarkeits- Ellipsoiden. Von besonderem Interesse ist dessen Volumen, das als Manipulierbarkeitsmaß[16] μ bezeichnet wird:

$$\mu = \sqrt{\left|\mathbf{J}_t \mathbf{J}_t^T\right|}. \tag{2.4}$$

Es beschreibt den Abstand zur nächsten Singularität. Das Manipulierbarkeitsmaß berücksichtigt in dieser Definition keine Gelenkwinkelbegrenzungen. Eine Erweiterung mit dieser Eigenschaft wird von (Abdel-Malek u. a., 2004) vorgeschlagen. Die Methoden eignen sich vor allem zum Entwurf von Roboterkinematiken.

Im dreidimensionalen kartesischen Raum können Abtastverfahren eingesetzt werden. (Guilamo u. a., 2005) benutzt die Vorwärtskinematik, um abgetastete Manipulatorkonfigurationen ins kartesische abzubilden und dadurch den erreichbaren Arbeitsraum abzutasten. Für jedes er-

[15] engl.: manipulability ellipsoid
[16] engl.: manipulability measure

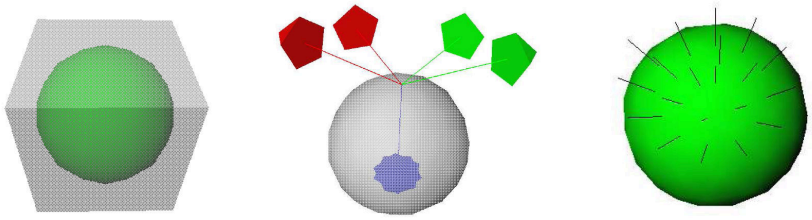


Abb. 2.4.: Erzeugung der Anrückrichtungen der Erreichbarkeitskarte: Links: Einbettung der Kugel in den einschließenden Würfel. Mitte: Zwei unterschieliche Frames in einem Punkt auf der Kugeloberfläche, die sich durch ihre Rotation um die z-Achse des Frames unterscheiden. Rechts: Kugel mit einigen visualisierten Anrückrichtungen. Quelle:(Zacharias u. a., 2007)

reichbare Voxel wird die erzeugende Konfiguration zusammen mit einem Priorisierungsmaß gespeichert. Abtastverfahren auf Basis der Vorwärtskinematik stellen jedoch die Erreichbarkeit des Arbeitsraums verzerrt da. In der Nähe von Singularitäten werden besonders viele Sample erzeugt (Zacharias u. a., 2007), zum Manipulieren sind diese Posen jedoch schlechter geeignet.

(Zacharias u. a., 2007) schlägt die *Erreichbarkeitskarte*[17] vor, eine Arbeitsraumbeschreibung basierend auf von Abtastung und inverser Kinematik. Der Arbeitsraum wird in gleichgroße Würfel unterteilt. In jeden Würfel wird eine Kugel eingepasst, die den Würfel an jeder Seite in genau einem Punkt berührt. Ihr Mittelpunkt sei mit M bezeichnet. Auf der Oberfläche werden nun n Punkte P gleichverteilt festgelegt. Für jeden Punkt P existiert eine Menge von Frames F, die ihren Ursprung in P haben und deren Z-Achse entlang des Vektors $\overrightarrow{PM}$ ausgerichtet ist. Sie unterscheiden sich lediglich in ihrer Rotation um die Z-Achse des Frames. Dies ist in Abb. 2.4 veranschaulicht. Der Punkt P bzw. der dazugehörige Frame F ist erreichbar, wenn es eine Rotation um die Z-Achse des Frames gibt, sodass mittels der inversen Kinematik eine Armkonfiguration gefunden wird, für die der Tool Center Point des Arms mit F übereinstimmt. Zur Bestimmung dieser Bedingung wird die Rotation um die Z-Achse weiter diskretisiert. Die Kugel mit den erreichbaren Richtungen wird als Erreichbarkeitskugel bezeichnet. Eine Erreichbarkeitskugel für den KUKA Leichtbauarm ist in Abb. 2.5 visualisiert. Für jede Kugel und jeden abgetasteten Würfel lässt sich damit der Erreichbarkeitsindex definieren, als Anteil der erreichbaren Punkte P auf der Kugeloberfläche an der Gesamtmenge aller Punkte: $\frac{|\{P:P \text{ ist erreicbar}\}|}{N}$.

Aufgrund der Verwendung der inversen Kinematik bringt das Erstellen der Erreichbarkeitskarte (abhängig von der Auflösung) einen großen Rechenaufwand mit sich. Dieser ist jedoch für die Verwendung unerheblich, da die Berechnung offline erfolgen kann und das Ergebnis in einer Datenbank gespeichert werden kann.

[17] engl.: reachability map. In älteren Publikationen, insbesondere in (Zacharias u. a., 2007) wird sie als capability map bezeichnet.

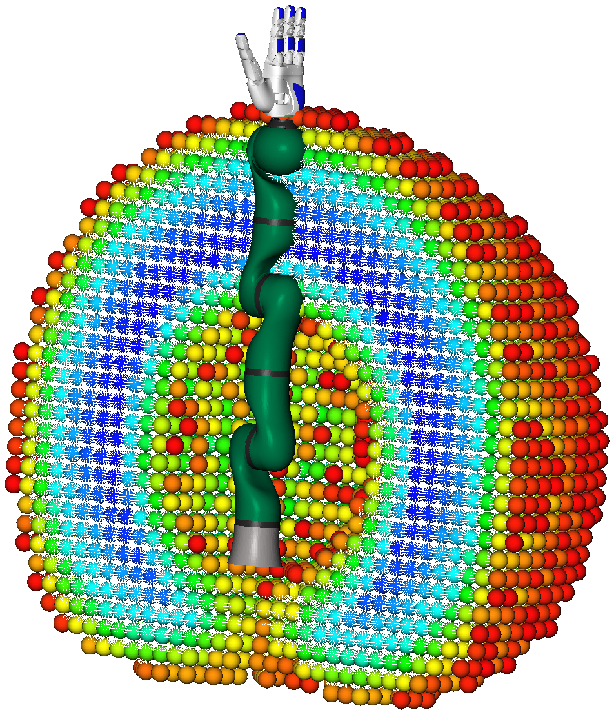


Abb. 2.5.: Schnitt durch die Erreichbarkeitskarte des KUKA Leichtbauroboters (LBR) entlang der X-Y-Ebene. Die Farben zeigen den Erreichbarkeitsindex an. Rote Farbe kennzeichnet eine schlechte Erreichbarkeit, blau eine sehr gute. Weiße Linien an den Kugeln zeigen eine erreichbare Richtung an.

2.4. Planungsverfahren im Kontinuierlichen

Symbolisch geplante Aktionen müssen auf ausführbare Trajektorien konkretisiert werden. Planungsalgorithmen für die in dieser Arbeit relevanten Aufgaben der kollisionsfreien Bewegungsplanung und Greifplanung sind im Stand der Technik hinreichend gut erforscht. Diese Arbeit liefert keinen Beitrag zu dem Forschungsgebiet, daher werden in den folgenden beiden Abschnitten lediglich die verwendeten Verfahren skizziert.

2.4.1. Bewegungsplanung

Ein symbolischer Planer kann abstrakte Aktionen wie „Fahre zu Greifpose” erstellen. Jedoch hat der symbolische Planer weder eine Modellierung von Hindernissen noch existieren auf seinem Abstraktionsniveau Trajektorien. Das Abbilden der symbolischen Aktion auf eine kontinuierliche Trajektorie muss daher von einem Planer im Kontinuierlichen durchgeführt werden. Bewegungsplaner lösen genau diese Aufgabe. Sie sind bereits gut erforscht. Einen Überblick geben etwa: (López u. a., 2008; Kavraki u. LaValle, 2008; LaValle, 2006).

Algorithmus 2.2 Probabilistic Roadmap Method aus (López u. a., 2008)

```
1:  function ERSTELLE_RPM
2:      n ← 0, V ← ∅
3:      while n < N do                          ▷ N: Anzahl zu erzeugende Sample
4:          q ← erzeugeKonfigurtaion()
5:          if q ∈ F then                       ▷ F: Freiraum
6:              U ← findeNachbarn(q)
7:              V ← V ∪ q
8:              for all u ∈ U do
9:                  if verifizierePfad(q,u) then
10:                     E ← E ∪ (q,u)
11:                 end if
12:             end for
13:             n ← n + 1
14:         end if
15:     end while
16: end function
```

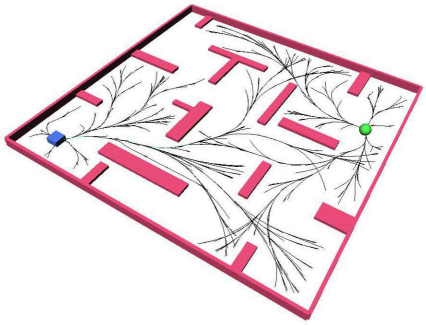


Abb. 2.6.: Zwei mit Rapidly Exploring Random Tree erstellte Explorationsbäume eines rechteckigen Objektes in einem Labyrinth. Aus (LaValle u. Kuffner, 1999).

Ein verbreiteter Ansatz ist die Probabilistic Roadmap Methode (RPM) (Kavraki u. a., 1996). Sie verwendet einen mehrstufigen Algorithmus zur Bewegungsplanung. Im ersten Schritt wird ein Graph von Konfigurationen aufgebaut, die Roadmap. Knoten in der Roadmap repräsentieren Konfigurationen, eine Kante zwischen zwei Konfigurationen existiert, wenn diese direkt voneinander kollisionsfrei erreichbar sind (z.B. bei linearer Interpolation der Gelenkwinkel). Zum Erstellen des Graphen wird Algorithmus 2.2 verwendet. Konfigurationen werden nach einer gegebenen Verteilung zufällig im Raum erzeugt. Die Erreichbarkeit wird für ihre bereits im Graph befindlichen Nachbarn berechnet. Nachbarn können z.B. durch ein k-Nearest Neighbour Verfahren bestimmt werden. Zur Bestimmung der Kollisionsfreiheit wird für jeden Pfad ein Kollisionsprüfer angefragt. Das Verfahren eignet sich besonders, wenn die Roadmap wiederverwendet werden kann, wenn also Anfragen für unterschiedliche Trajektorien in einer statischen Umgebung gestellt werden. Ein weiterer Vorteil ist, dass das Sampling im Konfigurationsraum stattfindet, die potentiell rechenintensive inverse Kinematik also nicht berechnet werden muss.

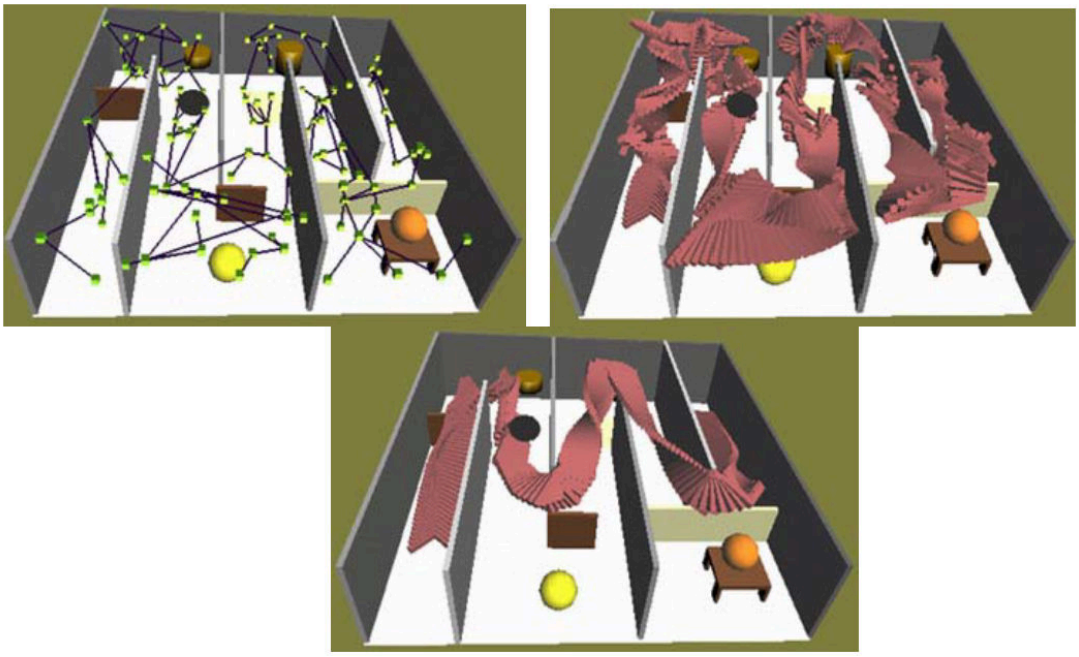


Abb. 2.7.: Visualisierung der Hauptschritte der Probabilistic Roadmap Method aus (López u. a., 2008). Oben links ist die Roadmap dargestellt. Sie ist das Ergebnis des Algorithmus 2.2. Oben rechts ist eine Trajektorie für eine konkrete Anfrage zu sehen, wie sie aus der Roadmap generiert wurde. Unten in der Mitte ist die geglättete Trajektorie dargestellt, die das Ergebnis der Planung ist.

Kann die Roadmap zwischen Anfragen nicht wiederverwendet werden, bieten sich ähnliche Verfahren zum Erstellen des Graphen an, etwa *Rapidly Exploring Random Trees* (RRT) (LaValle, 1998)). Sie können den Suchraum schneller abdecken und für eine Start—Ziel Planung optimiert werden. Ein RRT für eine Labyrinth–Szene ist in Abb. 2.6 dargestellt.

Der zweite Schritt der RPM behandelt eine konkrete Anfrage einer Trajektorie. Dazu wird eine direkte Verbindung von der Start- und Zielkonfiguration zu einem Knoten der Roadmap gesucht. Sind diese kollisionsfrei vorhanden, werden Start– und Zielkonfiguration, sowie die verbindende Kante zur Roadmap hinzugefügt. Es wird dann eine Verbindung im Graphen vom Start– zum Zielzustand gesucht. Diese enthält nur kollisionsfreie Kanten und damit einen kollisionsfreien Pfad, der Start und Ziel verbindet.

Die Schritte der RPM sind in Abb. 2.7 dargestellt. Es ist zu erkennen, dass die von der Suche erzeugte Lösung noch viele unnötige Bewegungen enthält. Um diese zu eliminieren, wird im letzten Schritt das Ergebnis geglättet.

In dieser Arbeit wird auf das weitverbreitete, auf der Probabilistic Roadmap Methode basierende, *Motion Planning Kit* (MPK)(MPK, 2006; Sánchez u. Latombe, 2003) zurückgegriffen.

2.4.2. Greifplanung

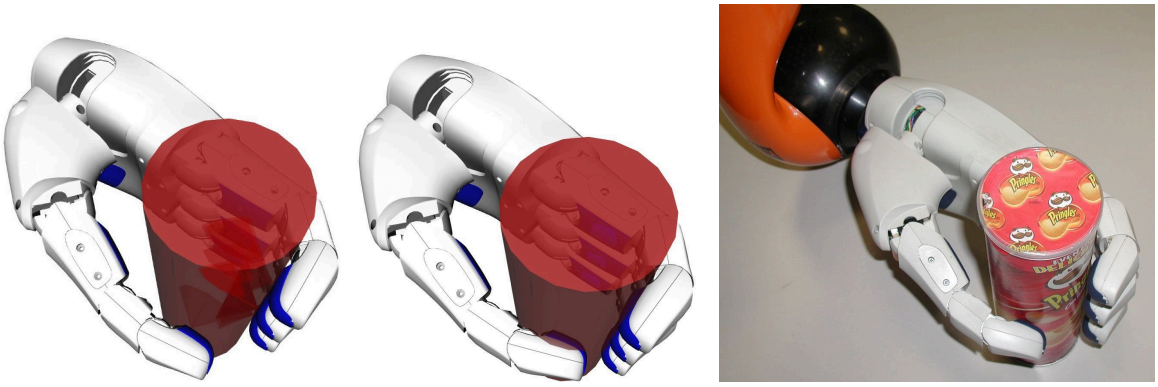
Zum Abbilden von Griffen auf kontinuierliche Trajektorien existiert ein gut erforschter Stand der Technik (Prattichizzo u. Trinkle, 2008; Kasper, 2013). Diese Arbeit verwendet das Verfahren von (Xue u. a., 2007, 2009, 2012), welches wiederum auf *GraspIt* von (Miller u. Allen, 2004) basiert.

GraspIt erzeugt Griffe durch simuliertes Schließen eines Greifers. Dazu wird der Greifer in eine vordefinierte „Preshape" Konfiguration gebracht. Der Greifer wird relativ zum Objekt positioniert, sodass die Konfiguration kollisionsfrei ist und sich das Objekt zwischen den Fingern des Greifers befindet. Nun werden die Finger unabhängig voneinander geschlossen, bis sich jeder Finger im Kontakt mit dem Objekt befindet. Objekt– und Greiferposition bleiben dabei unverändert. Die Kollision wird mittels eines Kollisionserkenners detektiert. Die sich ergebenden Gelenkwinkelkonfigurationen und Kontaktpunkte bestimmen den geplanten Griff. Durch unterschiedliche Preshape–Konfigurationen und Greiferpositionen können unterschiedliche Griffe für ein Objekt geplant werden. Für erstellte Griffe wird dann ein Qualitätsmaß berechnet, z.B. die größte Kugel im „Grasp-Wrench"-Raum (Kirkpatrick u. a., 1992). Aufgrund des Qualitätsmaßes können ungeeignete Griffe verworfen werden.

Das Erzeugen der Greiferposition und der Preshape–Konfigurationen basiert im GraspIt Simulator auf Expertenwissen. Der Experte wird dabei mit einer grafischen Oberfläche unterstützt. Weiter gibt es vordefinierte Greiferpositionen für primitive Objekte. Für die Greiferposition existieren auch Ansätze für automatische Lösungen. So wird im genutzten Ansatz von Xue auf eine Dekomposition der Objekte in Superquadriken zurückgegriffen (Goldfeder u. a., 2007). Für die modellierten Superquadriken sind geeignete Greiferposen vorgegeben, die damit auch für das Objekt geeignet sind. Ein weiterer vielversprechender Ansatz ist die Nutzung der medialen Achsen zur Bestimmung der Greiferposition (Przybylski u. a., 2011).

Das Greifplanungsverfahren von Xue basiert auf dem GraspIT Ansatz aus (Miller u. Allen, 2004). Vordefinierte Vorformen der Hand werden in der geometrischen Simulation um das Objekt positioniert. Geeignete Anrückpositionen und Richtungen werden aus einer Approximation der Objektgeometrie mittels Superquadriken erzeugt. Die Finger werden in Richtung des Objektes geschlossen. Durch eine Kollisionserkennung werden die Kontaktpunkte bestimmt. In einem Optimierungsschritt wird der Griff aufgrund einer Bewertungsfunktion seiner Stabilität angepasst. Die Nachgiebigkeitsregelung der Hand wird ausgenutzt, um mittels angepasster Fingerkonfigurationen definierte Kräfte auf das Objekt auszuüben. Ein Ergebnis ist in Abb. 2.8 dargestellt.

Die Menge der Griffe für ein Objekt entsteht durch Verwendung unterschiedlicher Vorformen der Hand und unterschiedlicher Anrückpositionen. Weitere Maßnahmen verbessern die Anwendbarkeit der Griffe. So wird die Tischannahme berücksichtigt, indem für tragende Ebenen



(a) (b) (c)

Abb. 2.8.: Adaption eines Griffes. In (a) ist der geplanter Griff dargestellt. (b) visualisiert die kommandierten Gelenkwinkel der Finger, die mittels Impedanzregelung und Widerstand des Objektes bei der realen Ausführung eine definierte Kraft ausüben. Abb. (c) zeigt die reale Ausführung. Quelle:(Xue u. a., 2012)

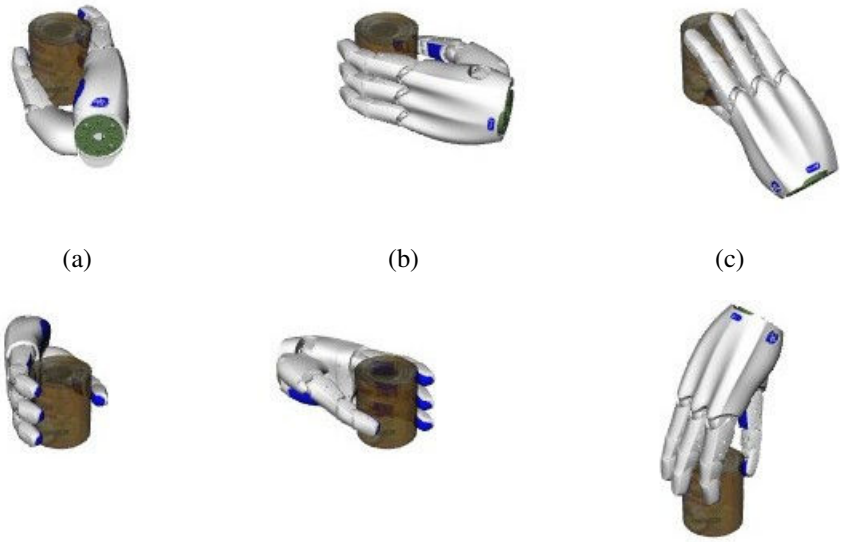
in stabilen Lagen Hindernisobjekte simuliert werden, die eine Kollision des erzeugten Griffs mit dem Tisch verhindern.

Einige Griffe für ein zylinderförmiges Objekt sind in Abb. 2.9 visualisiert. Ein durchschnittliches Objekt der KIT Objektdatenbank besitzt 23 Griffe (Median). Diese sind in einer Griffdatenbank gespeichert und können aus dieser mit vernachlässigbaren Zeitaufwand abgerufen werden. Für die Bestimmung der Greifbarkeit ist ein Objekt dann greifbar, wenn mindestens einer der Griffe des Objektes ohne Kollisionen mit Hindernissen ausgeführt werden kann.

2.5. Physiksimulation

Physiksimulationen, wie sie in dieser Arbeit verwendet werden, simulieren die Dynamik einer Szene von Objekten. Objekte sind dabei modelliert durch Eigenschaften aus Abschnitt 2.3.1, insbesondere geometrische Modelle, Masse und Reibung. In dieser Arbeit werden die Objekte als Starrkörper angenommen, lediglich der Manipulator selbst hat kinematisch modellierte Gelenke. Als weiterer, globaler Aspekt wird der Gravitationsvektor der Szene berücksichtigt.

Es werden Geschwindigkeiten und Impulse der Objekte generiert, ebenso wie Kontaktpunkte und Kräfte zwischen Objekten. Die Bewegung der Objekte wird als Differenzialgleichung beschrieben. Die Physiksimulation berechnet mittels numerischer Verfahren das zeitabhängige Verhalten des Systems. Um diese anwenden zu können, findet eine zeitliche Diskretisierung statt. Aufgrund dieser können die Ergebnisse der Physiksimulation lediglich eine Approximierung sein.



(a) (b) (c)

(d) (e) (f)

Abb. 2.9.: Visualisierung von sechs unterschiedlichen Griffen für das Sauerkrautdosenobjekt. Insgesamt enthält die Griffdatenbank 59 Griffe für das Objekt. Quelle: KIT Objektdatenbank (Kasper u. a., 2012)

In dieser Arbeit wird die Implementierung der Bullet Physics Engine (Bullet, 2014) verwendet. Sie erfüllt die notwendigen Anforderungen an simulierten Eigenschaften, Performance und Nutzbarkeit. Es wird eine empirisch ermittelte Parameteranpassung an verschiedene Problemstellungen benötigt.

2.6. Support Vector Machines

Support Vector Machines[18] (SVM) sind ein Verfahren zur Klassifikation von Datenpunkten in zwei Mengen (Schölkopf, 1999; Cortes u. Vapnik, 1995; Duda u. a., 2001). Ihre Anwendung erfolgt in zwei Phasen: In einer Trainingsphase wird aus einer Menge von Datenpunkten mit bekannter Klassenzugehörigkeit eine Parametrierung der SVM gelernt. In der Klassifikationsphase werden neue Datenpunkte durch die SVM klassifiziert, d. h. ihrer vermuteten Klasse zugeordnet.

Die Klassifikation wird in einem hochdimensionalen Raum durch eine separierende Hyperebene durchgeführt. In der Trainingsphase wird diese als Lösung des folgenden Optimierungsproblems für die Menge der Datenpunkte $\mathbf{x}_i$ mit der Klassenzugehörigkeit $y_i \in \{-1,1\}$ bestimmt.

[18] dt.: Stützvektormaschinen, die Bezeichnung ist jedoch ungebräuchlich.

Es wird das hier zuerst für den Fall linear trennbarer Trainingsmengen aufgestellt.

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} \\ \text{unter der Nebenbedingung} \quad & y_i(\mathbf{w}^T\Phi(\mathbf{x}_i)+b) \geqslant 1, \\ & \forall i. \end{aligned} \tag{2.5}$$

Dabei beschreibt der Normalenvektor $\mathbf{w}$ und das Bias b die resultierende Hyperebene, Φ die Abbildung in den hochdimensionalen Raum.

Als Ergebnis der Optimierung werden $\mathbf{w}$ und b so gewählt, dass die resultierende Hyperebene den Abstand zu den nächstliegenden Datenpunkten maximiert. Damit wird die Hyperebene nicht von der Gesamtmenge der Trainingsdaten bestimmt, sondern lediglich von einer kleinen Menge nah an der Trennhyperebene liegenden Punkten, den Stützvektoren. Dies führt zu einer erheblichen Datenreduktion gegenüber den Trainingsdaten sowie einer erhöhten Robustheit gegenüber neuen Datenpunkten, die sich zwischen den Trainingsmengen befinden bzw. gegenüber Überanpassung.

Im Fall nicht linear trennbarer Mengen werden die Hilfsvariablen ξ_i eingeführt, die die Fehlklassifikation (möglichst weniger) Datenpunkte erlauben:

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{1=1}^{l}\xi_i \\ \text{unter der Nebenbedingung} \quad & y_i(\mathbf{w}^T\Phi(\mathbf{x}_i)+b) \geqslant 1-\xi_i, \\ & \xi_i \geqslant 0, i = 1,...,l. \end{aligned} \tag{2.6}$$

Der Faktor C erlaubt die Gewichtung der Minimierung der Variablen ξ_i, die die Korrektheit der Klassifikation darstellt, gegenüber der Minimierung von $\mathbf{w}^T\mathbf{w}$, was die Größe des Trennbereiches repräsentiert.

Damit die Menge der Datenpunkte linear separierbar ist, muss sie durch die Funktion Φ in einen geeigneten höherdimensionalen Raum transformiert werden. Nach (Duda u. a., 2001) existiert eine solche Transformation immer, wenn der Raum, in dem klassifiziert wird, genügend Dimensionen hat. Dort kann das Optimierungsproblem auf ein duales Problem abgebildet werden, für das eine Lösung bestimmt werden kann, die mit der des primären Problems identisch ist. Um dabei die aufwändige Berechnung im hochdimensionalen Raum zu vermeiden, wird der sogenannte „Kernel Trick" angewendet. Er nutzt aus, dass die einzige notwendige Berechnung dort das Skalarprodukt $K(\mathbf{x}_i,\mathbf{x}_j)$ ist, das bei geeigneter Wahl von Φ auch im ursprünglichen Raum bestimmt werden kann:

$$K(\mathbf{x}_i,\mathbf{x}_j) = \Phi(\mathbf{x}_i)^T\Phi(\mathbf{x}_j).$$

Ein Kernel, der nichtlineare Trennung zwischen Klassen leisten kann, ist die *Radial Basis Function* (RBF), (Chang u. Lin, 2001). Er wird definiert durch:

$$K(\mathbf{x_i},\mathbf{x_j},) = e^{-\gamma||\mathbf{x_i}-\mathbf{x_j}||^2}. \tag{2.7}$$

Die Klassifikation unbekannter Punkte ergibt sich aus der Lösung des dualen Problems durch die Funktion:

$$y(x) = \operatorname{sgn}\left(\sum_{i=0}^{n} \alpha_i y_i K(x_i, x) + b\right) \tag{2.8}$$

mit n als Anzahl der Stützvektoren.

2.7. Impedanzregelung

Ein Roboter, der in einer nicht exakt bekannten Umgebung agiert, benötigt Methoden, Manipulationsaktionen an diese anzupassen. Auf unteren Ebenen, im Bereich von Kraft- oder Positionsregelung, können Ansätze, die dem Roboter ein nachgiebiges Verhalten ermöglichen, dazu verwendet werden. Ein solcher Ansatz ist die Impedanzregelung.

Ihr Verhalten wird durch folgende Gleichungen beschrieben:

$$\begin{aligned} M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) &= \tau + \tau_{ext} \\ B_\theta\ddot{\theta} + D_\theta\dot{\theta} + K_\theta(\theta - \theta_s) + \tau &= 0 \end{aligned} \tag{2.9}$$

Im statischen Fall mit $\dot{\theta} = \ddot{\theta} = 0$ reduziert sich (2.9) zu:

$$\theta_s = \theta + K_\theta^{-1}\tau \tag{2.10}$$

wobei $M(q)$ die Trägheitsmatrix des Manipulators in der Konfiguration q ist und C den Coriolis-Effekt modelliert. g stellt Momente in den Gelenken, die durch die Gravitation entstehen, dar. Bei B, D und K handelt es sich um die virtuelle Masse, Dämpfung und Steifigkeit des Systems. θ_s ist die resultierende Gelenkkonfiguration, θ die kommandierte und τ ein externes Moment. Die Gleichung kann in den kartesischen Raum übertragen werden.

2.8. Fazit

Ausgehend von Planungsverfahren in der Robotik wurde der Stand der Technik darauf untersucht, inwieweit Lösungen für die gefundenen Teilaufgaben existieren. Als Ergebnis wurde im

Bereich der Konkretisierung ein hinreichender Fundus an bekannten Planungsverfahren und Modellen (s. Abschnitt 2.4) im kontinuierlichen Raum identifiziert. Dies gilt ebenso für Planungsverfahren im Symbolischen. Symbolische Planer, wie in Abschnitt 2.2 beschrieben, stehen als Bibliotheken zur Verfügung. Zur Beschreibung der Planungsdomäne existieren Modelle und Sprachen (s. Abschnitt 2.1). Bei Weitem nicht so tief gehend erforscht ist jedoch die Frage, welche Aspekte einer Manipulationsaufgabe in diesen Modellen beschrieben werden und wie dies geschieht. Darauf wird im nächsten Kapitel eingegangen.

3. Stand der Technik

Im vorigen Kapitel wurden die verwendeten Grundlagen dieser Arbeit beschrieben. Im Gegensatz dazu wird in diesem Kapitel der Stand der Technik der Methoden zur Symbolisierung, Handlungserzeugung und Ausführungsüberwachung beschrieben, zu denen diese Arbeit einen Beitrag leistet. Der Stand der Technik wird dabei anhand der Anforderungen dieser Arbeit bewertet und es werden zu schließende Lücken identifiziert.

Der Fokus der Betrachtung liegt auf Methoden, die für Manipulation und symbolische Planung geeignet sind. Es werden darüber hinaus weitere Felder betrachtet. So existiert etwa im Bereich der Navigation für mobile Roboter ein sehr fortgeschrittener Stand der Technik, sowohl bei der Modellierung als auch der Planung. Werden Methoden aus diesen beschrieben, so ist dies mit einer Analyse der Eignung bzw. Übertragbarkeit für die Aufgabenstellung dieser Arbeit verbunden. Folgende Punkte werden, soweit in den Abschnitten anwendbar, untersucht:

Manipulation Die Verfahren müssen zu Manipulationsaufgaben kompatibel sein. Es dürfen keine nicht übertragbaren Eigenschaften der spezifischen Anwendung ausgenutzt werden.

Neue Aktionen Die Verfahren sollen in der Lage sein, Sequenzen von Aktionen zu generieren, welche die Szene im Sinne eines angestrebten Zieles verändern. Besonders das Erzeugen neuer Aktionen in der Sequenz ist von Interesse.

Wissenstransfer Es wird untersucht, ob es Wissenstransfer zwischen den Abstraktionsebenen gibt, der über den Austausch von Positionen und dem Auslösen von Aktionen hinausgeht.

Adaption Die Verfahren müssen in der Lage sein, auf strukturell sehr unterschiedlichen Szenen Ergebnisse zu liefern.

Onlinefähigkeit Die Laufzeiten der Verfahren müssen eine Online-Nutzung ermöglichen. Benutzerinteraktion oder langwierige Rechenoperationen, die für jede Szene durchgeführt werden müssen, sind nicht akzeptabel.

Wirklichkeit Die Anbindung an die Wirklichkeit muss vorhanden sein. Es wird betrachtet, inwiefern die Methoden mit verrauschten Sensordaten und fehlschlagenden Aktionen umgehen können.

Evaluiert Ebenfalls betrachtet wird, ob ein Verfahren ausschließlich theoretisch vorgestellt oder an einem Roboter evaluiert wurde und in welchem Szenario das geschah.

3.1. Symbolisierung

In diesem Abschnitt werden Methoden zur Abbildung relevanter Aspekte eines konkreten Zustands in eine symbolische Beschreibung untersucht. Dieser Prozess wird als Symbolisierung bezeichnet.

Einen sehr generellen Ansatz, um Symbolverankerung[1](Harnad, 1990) aus dem Kontinuierlichen ins Symbolische zu definieren, schlägt (Tenorth u. a., 2010) vor. Numerisch beschriebenen Eigenschaften des kontinuierlichen Zustands werden durch beliebige Funktionen auf einzelne Symbole abgebildet. Diese Perspektive der Symbolisierung liegt auch dieser Arbeit zugrunde. Daraus definieren die im Folgenden beschriebenen Ansätze gerade die Funktionen zur Symbolisierung.

3.1.1. Symbolisierung des Arbeitsraums

Das Erzeugen symbolischer Modelle aus kontinuierlich dargestellten Umweltmodellen tritt im Bereich der mobilen Robotik bzw. der Navigation auf. Dort hat man auf der einen Seite kontinuierliche, metrische Karten, die mit Sensoraufnahmen der Welt abgeglichen werden. Auf der anderen Seite bieten sich semantische Darstellungen an. In ihnen sind relevante Orte modelliert, die für die Pfadplanung entsprechend ihrer Erreichbarkeit miteinander verbunden sind. Sie können semantisch mit Namen verknüpft werden, um einem Benutzer ihre Spezifikation zu vereinfachen (z.B. „Telefon in der Eingangshalle”), oder mit weiteren Informationen, wie sich ein Fahrzeug an ihnen verhalten soll (z.B. „Sicherheitsabstand halten”).

Eine hierarchische Kartendarstellung, die *räumliche semantische Hierarchie*[2], wird von (Kuipers, 2000) vorgeschlagen. Sie kennt Sensor-, Steuerungs-, metrische, topologische und kausale Abstraktionsebenen. Ein ähnliches Modell verwendet (Pronobis u. a., 2009). Es ist in Abb. 3.1 dargestellt. Auf der untersten Ebene ist die *metrische Karte*. Sie entspricht einem kontinuierlichen, zweidimensionalem geometrischen Modell der Umgebung und kann während der Ausführung durch simultanes Lokalisieren und Kartieren[3] (SLAM) gewonnen werden, oder sie ist a-priori bekannt. Darüber befindet sich die Navigationskarte. Sie enthält diskretisierte, für den Roboter erreichbare Orte. Sie wird bei der Exploration der Umgebung aufgebaut, indem periodisch angefahrene Orte der Karte hinzugefügt werden. Die Navigationskarte ist mit In-

[1] engl.: Symbol Grounding
[2] engl.: Spatial Semantic Hierarchy
[3] engl.: Simultaneous Localization and Mapping

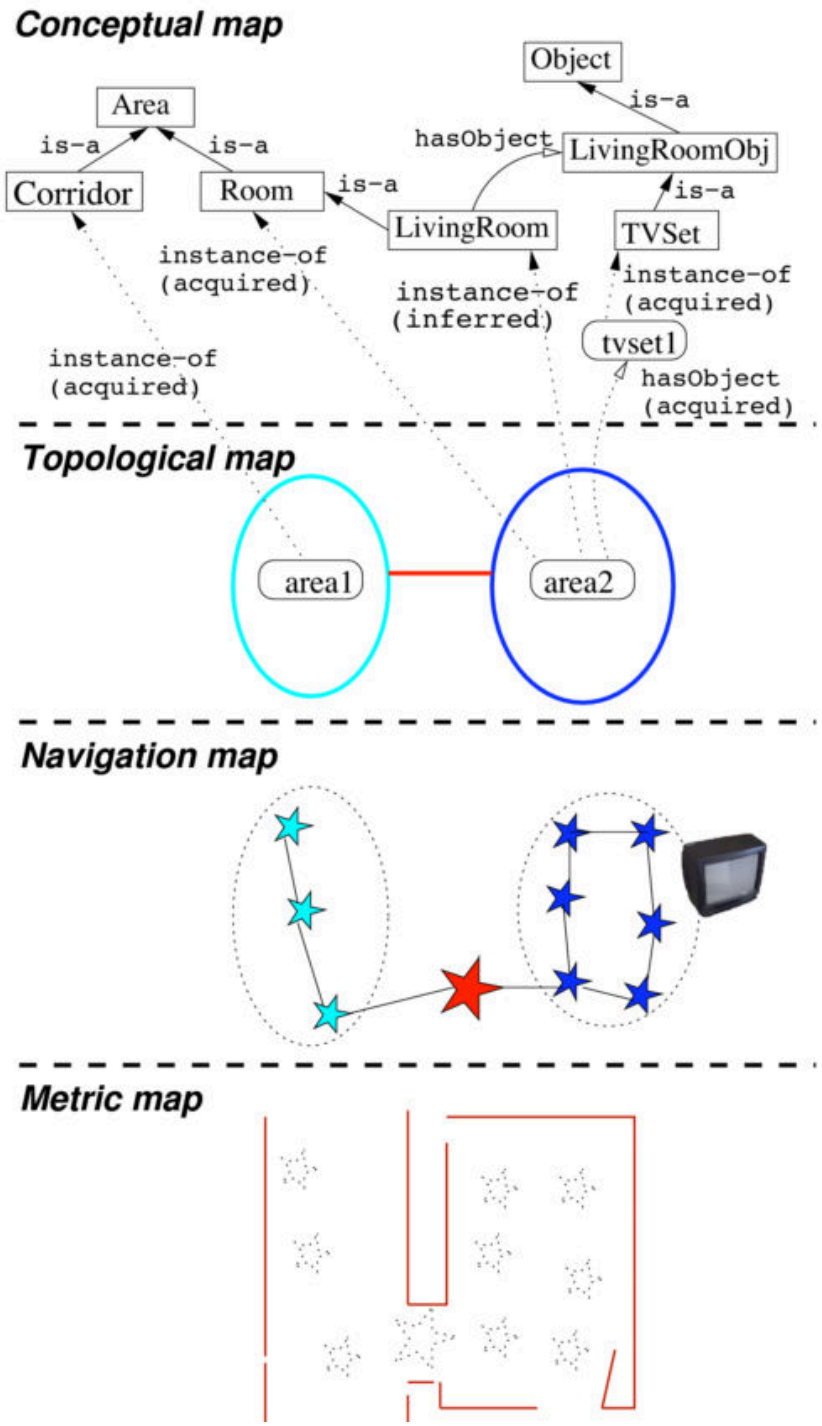


Abb. 3.1.: Semantisches Kartenmodell nach (Pronobis u. a., 2009). Auf der untersten, konkreten Ebene befindet sich die metrische Karte. Die Ebenen drüber sind symbolisch. Die Navigationskarte modelliert die Erreichbarkeit für den Roboter, die topologische Karte zusammenhängende Regionen (z.B. Räume).

formationen über Türen und Objekte annotiert. Die topologische Karte fasst Punkte zu Gebieten zusammen. Zur Erstellung wird eine Segmentierung der Navigationskarte anhand von Hindernissen durchgeführt. Diese Darstellung entspricht einer für Menschen natürlich wirkenden Form. Auf der obersten Ebene wird die konzeptuelle Karte erstellt. Sie geht über räumliche Aspekte hinaus und fasst Objekte in einer Ontologie zusammen.

Die Symbolisierungsverfahren in beiden Arbeiten basieren auf der Unterteilung der Umgebung in Räume durch unüberwindbare Hindernisse. Dieser Ansatz ist nicht auf Manipulationsaufgaben übertragbar. Als weiterer Ansatz zum Erstellen einer semantischen Karte wurde in (Diosi u. a., 2005) Benutzerinteraktion während einer Fahrt durch das zu kartierende Gebäude untersucht. In (Martínez Mozos u. a., 2007) kommt ein überwachtes Lernverfahren zum Einsatz. Es erfordert ebenfalls das Einbinden des Benutzers. Diese Ansätze sind nur praktikabel, wenn sich die Umgebung nach der Benutzerinteraktion nicht verändert, für variable Szenen sind sie nicht geeignet.

In der mobilen Manipulation interessiert neben der Unterteilung des Raumes die Fragestellung, an welchem Ort ein Manipulator platziert werden kann, um ein Objekt zu manipulieren.

(Stulp u. a., 2009) stellt dazu ein Lernen-basiertes Verfahren vor. Ein Objekt wird im Raum positioniert. Es werden zufällig verteilte Roboterpositionen um das Objekt generiert, dann wird durch Greifplanungsalgorithmen bestimmt, ob das Objekt von dieser Position greifbar ist. Dieser Schritt wird in der Simulation durchgeführt und für verschiedene Objektposen wiederholt. Es ergibt sich zu jeder Objektpose eine Menge von Roboterposen, von denen aus das Objekt greifbar sein kann. Mittels Support Vector Machines und einem Punktverteilungsmodell[4] (PDM) werden die Roboterposen in eine Darstellung transformiert, in der die Klassifikation einer Roboterpose abhängig von einer beliebigen, kontinuierlich gegebenen Objektpose berechnet werden kann. Es ergibt sich eine Bewertungskarte für Posen um das Objekt, die die Eignung der Roboterpose zum Greifen beschreibt. Aus dieser kann eine gut bewertete Pose ausgewählt werden. In (Fedrizzi u. a., 2009) wird das Verfahren zur Erstellung komplexer Pläne für einen mobilen Manipulator eingesetzt. Es werden Anfahraktionen erzeugt, die den Roboter in eine zum Greifen des Objektes geeignete Pose bringen.

Eine ähnliche Aufgabenstellung wie in (Stulp u. a., 2009) wird in (Aydin u. Nakajima, 1999) untersucht. Auch hier werden Orte und Konfigurationen zum Greifen von Objekten generiert. Der Ansatz ist jedoch rein zur Visualisierung ausgelegt. Die Konfigurationen werden nach Aussehen generiert. Ihre Funktionalität wird nicht untersucht. Auch berücksichtigt der Ansatz keine Kollisionen. Eine Übertragung auf robotische Manipulation ist daher nicht möglich.

Weitere Verfahren generieren Roboterposen zum Greifen von Objekten unter Berücksichtigung der Manipulatorkinematik und von Objekteigenschaften: In (Zacharias u. a., 2006) wird gezeigt,

[4] engl.: point distribution model

wie die Erreichbarkeitskarte eines Roboters verwendet werden kann, um Anrückpositionen zum Greifen eines Objektes zu erzeugen. (Berenson u. a., 2008) verwendet einen Sampling-basierten Planer um eine Menge von Griffkonfigurationen zu erzeugen. Aus diesen wird mit aufgabenspezifischen Bewertungsfunktionen ausgewählt.

3.1.2. Symbolisierung von Szenenrelationen

In diesem Abschnitt werden Methoden zum Symbolisieren von Relationen zwischen Objekten aus einem kontinuierlichen Modell untersucht. Solche Relationen werden in dieser Arbeit als *Szenenrelationen* bezeichnet. Sie beschreiben Beziehungen zwischen Objekten in einer Szene, die aus der relativen Lage der Objekte und deren physischen Eigenschaften wie Ausdehnung, Geometrie oder Masse entstehen. Es wird zwischen räumlichen und mechanischen Szenenrelationen unterschieden.

- *Räumliche Szenenrelation* beschreiben räumliche Beziehungen zwischen Objekten in einer statischen Szene. So eine Relation kann zum Beispiel „Bei", „Auf", „Hinter" oder „Rechts von" sein. Sie haben einen deskriptiven Charakter bezüglich der Struktur der Szene. Menschen tendieren dazu, diese Relationen zur Beschreibung der Anordnung einer Szene zu verwenden, da sie ihnen intuitiv zugänglich sind. In der deutschen und englischen Sprache (Clark, 1973) werden sie durch lokale Präpositionen ausgedrückt.
- *Mechanische Szenenrelationen* beschreiben zwischen Objekten wirkende Kräfte einer statischen Szene. Zu ihnen gehören unter anderen „Trägt", „Steht auf", „Stützt" und „Lehnt an". Sie charakterisieren damit einen Einfluss, den ein Objekt auf ein anderes hat und der sich insbesondere bemerkbar macht, wenn das Objekt bewegt wird. In der deutschen Sprache werden diese Relationen durch Verben beschrieben.

Eine besondere Relation stellt die „Auf" Relation dar. Sie beschreibt primär eine räumliche Beziehung, impliziert jedoch auch eine tragende Eigenschaft des unteren Objektes, also eine Kraft zwischen den beiden und damit eine mechanische Beziehung.

Räumliche Szenenrelationen Räumliche Relationen[5] werden in diversen Forschungsgebieten untersucht, insbesondere in der Psychologie (Sehen), der künstlichen Intelligenz (Räumliches Schließen) und in der Geographie (geographische Informationsysteme) (Papadias u. Kavouras, 1994). Ebendiese geben einen Überblick über Aufgeben mit Bezug zu räumlichen Relationen, diese sind:

- *Konstruktion* der Relationen aus uninterpretierten Daten (z. B. Perzeption)

[5] engl.: Spatial Relations

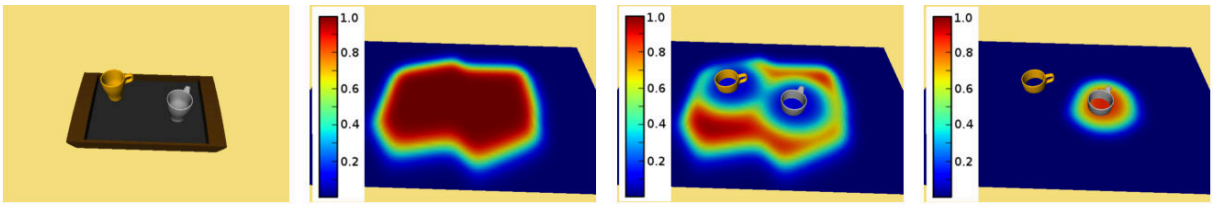


Abb. 3.2.: Modellierung geometrischer Relationen nach (Burbridge u. Dearden, 2012). Links: Initiale Umgebung mit Tablett und zwei Tassen. Mitte links: Wahrscheinlichkeit für eine Tasse *auf* dem Tablett, daneben Wahrscheinlichkeit für eine weitere Tasse auf dem Tablett, welche die anderen nicht berührt. Rechts: Wahrscheinlichkeit für eine Tasse, welche die andere berührt.

- *Transformation* der Relationen aus interpretieren Daten
- *Schließen* von neuem Wissen aus räumlichen Relationen

Im Rahmen der Symbolisierung interessiert hier der Bereich der Konstruktion.

(Freeman, 1975) analysiert die Modellierung räumlicher Aspekte. Die Arbeit legt dabei besonderen Wert auf die Charakterisierung der Eigenschaften der Relationen, sodass diese der Wahrnehmung des Menschen entspricht. Dazu werden Arbeiten aus der Psychologie herangezogen. Die Untersuchungen erfolgen auf zweidimensionalen Skizzen. Ein wesentliches Ergebnis ist, dass die menschliche Wahrnehmung räumlicher Relationen eher unscharf ist.

In der Bildverarbeitung will man räumliche Relationen von Objekten, die im Bild als Bereiche auftreten, aus Bildern ableiten. (Keller u. Wang, 1996) untersuchen dazu die Relationen „Links von", „Rechts von", „Über" und „Unter". Eine Möglichkeit zur Bestimmung ist die Betrachtung von Länge und Richtung der verbindenden Strecken charakteristischer Punkte der Bereiche. Diese Methode kann für Regionen und unscharfe Mengen erweitert werden. In der Arbeit wird eine Methode vorgeschlagen, die mit neuronalen Netzen binäre Bilder geometrisch einfacher Objekte (Punkte, Linen und L-Formen) Relationen zuordnet. Die Verfahren arbeiten auf binären, bereits segmentierten, zweidimensionalen Bildern. Sie sind also weit von echten Sensoraufnahmen entfernt.

Ein Verfahren, das Objekte einer dreidimensionalen Szene auf Rechteck-Algebra (Balbiani u. a., 1999) abbildet, wird in (Mansouri u. Pecora, 2013) vorgestellt. Rechteck-Algebra ist eine Erweiterung der Allen Relationen (vgl. Abschnitt 2.1.5 in den zweidimensionalen Raum. Diese wird genutzt, um die Lage von Objekten auf einer Ebene zueinander zu beschreiben.

(Burbridge u. Dearden, 2012) beschreiben ein Verfahren zum Lernen bi-direktionaler Abbildungen zwischen geometrischen Szenenbeschreibungen und symbolischen räumlichen Relationen. Mittels eines überwachten Lernverfahrens wird ein Gauß'scher Kerneldichteschätzer erstellt. Er modelliert die Wahrscheinlichkeit, dass ein 5-Tupel, bestehend aus der Größe der beiden Objekte, der relativen Verschiebung der Objekte und der Orientierung jedes Objektes, einer gegebenen Relation entspricht. Translationen und Rotationen werden dreidimensional dargestellt.

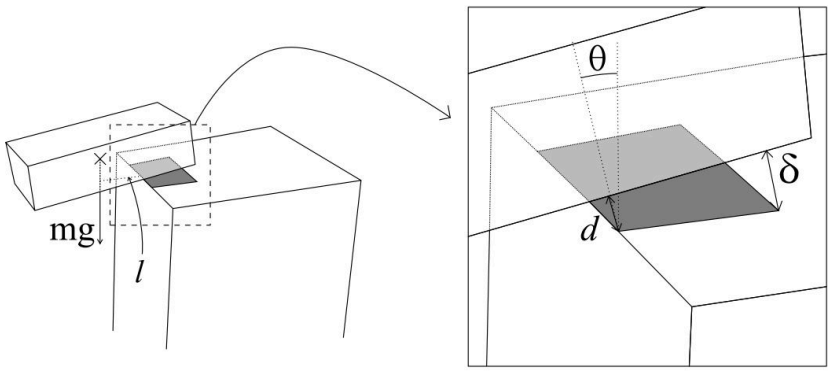


Abb. 3.3.: Merkmale zur Bestimmung der „Auf"-Relation aus (Sjöö u. a., 2010). l beschreibt den Abstand der Schwerpunkte der Objekte, d die Distanz zwischen den Oberflächen der Objekte. θ ist der Kontaktwinkel, also der Winkel zwischen der Normalen der Kontaktfläche und der vertikalen Achse.

Die erzeugten Relationen werden zur Evaluation in einem symbolischen Planer verwendet. Ein Roboter kann so eine symbolisch beschriebene Szenenkonfiguration durch Manipulieren der Objekte erzeugen. In der Veröffentlichung werden nur die Relationen „Bei" und „Auf" untersucht. Diese kommen der verwendeten Modellierung sehr entgehen, da für beide Relationen jeweils nur Abstand und Größe der Objekte relevant sind.

Im Rahmen dieser Arbeit wird Wissen benötigt, das über räumliche Relationen hinaus geht, da räumliche Relationen nicht die Auswirkungen von Veränderungen zwischen den beteiligten Objekten beschreiben.

Mechanische Relationen In diesem Abschnitt soll zuerst die „Auf" Relation betrachtet werden, wie sie in (Sjöö u. a., 2010) zur Umgebungsmodellierung für mobile Roboter verwendet wird. Der Autor selbst bezeichnet die Relation als räumlich, sowohl bei der Motivation als auch der Anwendung wird jedoch auf mechanische Effekte eingegangen.

In (Sjöö u. a., 2010; Sjöö u. Jensfelt, 2011) wird ein Verfahren beschrieben, das die „Auf"-Relation aus geometrischen Modellen generiert. Dafür werden für Objektpaare in Berührung folgende Merkmale erzeugt: Die Distanz zwischen den Oberflächen der Objekte d. Das ist der kürzeste Abstand bei sich nicht berührenden Objekten oder der längste Abstand bei sich überschneidenden Objekten. Durch das Zulassen eines Abstandes δ kann durch Sensorik bedingte Ungenauigkeit berücksichtigt werden. l beschreibt die horizontale Distanz der Schwerpunkte der Objekte. Zur Schwerpunktschätzung wird das geometrische Zentrum verwendet. θ ist der Kontaktwinkel. Das ist der Winkel zwischen der Normalen der Kontaktfläche und der vertikalen Achse. Eine Visualisierung ist in Abb. 3.3 zu sehen. Die Merkmale werden über Bewertungsfunktionen auf den Werteraum [0,1] abgebildet und darüber die Zugehörigkeit zur „Auf"-Relation berechnet. Die beschriebene Modellierung führt zu Uneindeutigkeiten bei Ob-

jekten, die aneinander lehnen. Diese sind vor allem widersprüchlich zur intuitiven Interpretation. Ein Objekt, das an einem anderen lehnt, kann je nach Winkel als auf diesem beschrieben werden oder nicht. Die evaluierte Anwendung ist eine Szenenbeschreibung, in der mittels „Auf"-Relation der Ort der Objekte beschrieben wird. Ein mobiler Roboter kann in der Umgebung navigieren, das Modell erstellen und die durch die „Auf"-Relation beschriebenen Objekte finden. In (Sjöö u. a., 2012) wird weiter eine „In"–Relation vorgestellt, mit analogen Methoden erkannt und zur Modellierung von Objektpositionen verwendet. Von (Pangercic u. a., 2010) wird ebenfalls die Nutzung der „Auf"-Relation für einen mobilen Manipulator vorgeschlagen. (Beetz u. a., 2012) verwendet eine Physiksimulation, um eine *Ist-Stabil* Eigenschaft für ein einzelnes Objekt zu erzeugen.

In (Sjöö u. Jensfelt, 2011) wird eine Physiksimulation eingesetzt, um Trainingsdaten zum Lernen mechanischer Relationen zu generieren. Dazu werden zufällig Szenen generiert und dann mittels Bewertungsfunktionen auf Kräften und dem Effekt des Entfernens von Objekten die „Stützt"-Relation erzeugt. Auf den erzeugten Trainingsdaten werden 93 unterschiedliche Merkmale für Objektpaare erzeugt. Diese werden mittels logistischer Regression klassifiziert. Letztlich liegt der Fokus auf dem Lernverfahren, dieses wird in Bezug auf simulierte Szenen evaluiert. Weder eine Eignung der mechanischen Simulationsmodelle in Bezug auf die beschriebenen Relationen, noch eine Anbindung an die Wirklichkeit werden untersucht.

(Kleer u. Brown, 1984) schlägt eine Sprache zur Beschreibung komplexer physikalischer Zusammenhänge vor. Diese soll eine qualitative Aussage über das Ergebnis des Zusammenspiels von Komponenten in einem physikalischen System erlauben. Als Beispiel wird ein Druckventil betrachtet. Das System wird von einem Experten modelliert. Zum Erzeugen nützlicher Ergebnisse muss es vom Benutzer aufgabenspezifisch ergänzt werden. Die erzeugten Ergebnisse bilden das Systemverhalten nicht in allen Aspekten korrekt ab.

3.1.3. Symbolisierung von Aktionen und deren Eigenschaften

(Choi u. Amir, 2009) schlagen eine Zerlegung des Arbeitsraums mittels eines Bahnplaners vor. Dazu werden Punkte im Raum verteilt. Existiert eine kollisionsfreie Trajektorie zwischen den Punkten, so wird eine Aktion zur symbolischen Menge der Aktionen hinzugefügt, die den TCP vom ersten zum zweiten Punkt bewegt. In einer Simulation können damit Planungsprobleme gelöst werden, die sowohl symbolisch definierte Ziele erfüllen als auch die Kollisionsfreiheit des Plans sicherstellen. Auch eine kinematische Simulation wird durchgeführt. Allerdings berücksichtigt das System keine Griffe, die weitere erreichbare Posen erforderlich machen würden. Vorstellbar wäre, Objektpositionen zu integrieren, aus denen TCP Positionen zum Greifen generiert werden können. Sind diese jedoch veränderlich, so müssen die Aktionen für jede Sze-

nenkonfiguration neu generiert werden, womit die Onlinefähigkeit des Ansatzes nicht gegeben ist.

3.1.4. Fazit

In diesem Abschnitt wurden Verfahren zur Symbolisierung unterschiedlicher Aspekte von Szenen und Aktionen betrachtet. Übergreifende Ansätze zur Anwendung in einem Planer sind im Stand der Technik nicht bekannt, für einzelne Teilaspekte existieren Verfahren. Ein Überblick über die Verfahren ist in Tab. 3.1 gegeben.

Verfahren	Anwendung	Manipulation	Neue Aktionen	Wissenstransfer	Adaption	Online	Wirklichkeit	Evaluiert
Semantische Karten	Verknüpfung von Ort und Beschreibung; Navigation	⊗		⊕	⊕	⊕	⊕	⊕
Bewertungskarten von Greifposen	Bestimmen von Roboterposen mobile Manipulation	⊕		⊖	⊖	⊕	⊕	⊕
Räumliche Relationen aus 2D Bildern	Bildverarbeitung, erzeugen einer verbalen Beschreibung	⊗		⊕	⊘	⊕	⊗	⊗
Lernen räumlicher Relationen	Zielbeschreibung für die Manipulation	⊕		⊕	⊘	⊕	⊕	⊕
Lernen mechanischer Relationen	Szenenbeschreibung für die Manipulation	⊕		⊕	⊘	⊕	⊕	⊕
Symbolisierung von Bewegungen	Aktionsbeschreibung für die symbolische Planung	⊘		⊕	⊗	⊘	⊗	⊗

Tab. 3.1.: Überblick über die diskutierten Verfahren im Bereich der Symbolisierung. ⊕ : gut geeignet; ⊘ mit Einschränkungen geeignet, ⊖ schlecht geeignet, ⊗ ungeeignet, leeres Feld: Kriterium kann nicht angewandt werden. Zur Bedeutung der Kriterien vlg. Anfang des Kapitels.

Im Bereich der Arbeitsraum-Symbolisierung ist vor allem der für die Navigation relevante Bereich der automatisch erzeugten semantischen Karten hervorzuheben. Dort bekannte Verfahren erfüllen viele Anforderungen dieser Arbeit, sind aber nicht auf die Manipulation zu übertragen. Ein weiterer bedeutender Bereich ist das Erzeugen von Roboterposen zum Greifen von Objekten. Dazu werden Bewertungskarten eingesetzt. Die bekannten Ansätze werden in dieser Arbeit weiterentwickelt, um den Arbeitsraum eines Manipulators beim Greifen von Objekten in Szenen mit hoher Variabilität zu beschreiben.

Bei räumlichen Szenenrelationen gibt es zum Einen Anwendungen aus dem Bereich der Bildverarbeitung bzw. des Bildverstehens. Diese arbeiten auf zweidimensionalen Daten und zum

Teil bereits vorsegmentierten Bilddaten. Sie sind nicht geeignet, Beschreibungen aus echten Sensoraufnahmen zu erzeugen, die eine die mechanische Semantik der Szene erhalten. Zum anderen gibt es Ansätze aus dem Bereich der Robotik (Burbridge u. Dearden, 2012), die zur Planung nutzbare räumliche Beschreibungen erzeugen. Solche Ansätze wurden bereits auf echten Robotern evaluiert. Im Rahmen dieser Arbeit wird hier kein weiterer Forschungsbedarf gesehen.

Dagegen existiert weiterer Forschungsbedarf bei den mechanischen Szenenrelationen. Die Arbeiten von Sjöö liefern vielversprechende Ergebnisse, jedoch nicht in der Modellierungstiefe, wie sie für die autonome Planung von Manipulatiossequenzen auf unbekannten Szenen benötigt werden.

Ein ähnliches Bild ergibt sich bei der Symbolisierung von Aktionen. Der in (Choi u. Amir, 2009) vorgeschlagene Ansatz ist auf wenige Aktionstypen beschränkt und kann sich nicht online an neue Szenen anpassen. In den beiden im vorigen aufgeführten Bereichen wird der Stand der Technik daher in dieser Arbeit weiterentwickelt.

3.2. Handlungsgenerierung

In diesem Abschnitt werden Verfahren untersucht, die Aktionen für Serviceroboter erzeugen. Dabei liegt der Fokus insbesondere auf Ansätzen, die mehrere Aktionen für mehrere bewegliche Objekte erzeugen. Dieses Problem wird auch als Manipulationsplanung (LaValle, 2006), bzw. wenn nur der Raum der Objekte und kein Manipulator betrachtet wird, als Rearrangier-Planung (Rivlin, 1998) bezeichnet. Es ist PSPACE hart (Wilfong, 1988) und damit im allgemeinen nicht in für diese Arbeit akzeptablen Zeitschranken lösbar. Stattdessen müssen Einschränkungen definiert und Heuristiken eingesetzt werden.

3.2.1. Erweiterung von Aktionsplanungsverfahren

Der aSyMov Planer (Gravot u. a., 2005; Cambon u. a., 2004, 2009) führt den Begriff der *Positionen* ein, um einen symbolischen mit einem geometrischen Planer zu verbinden. Eine Position ist eine Repräsentation einer geometrischen Pose im Symbolischen; sie kann also beispielsweise die Bedeutung „Ort, von dem Objekt A gegriffen werden kann" haben. Eine symbolische Position wird mit einer Menge von Konfigurationen annotiert, die ein Roboter einnehmen kann, um im Geometrischen die Bedeutung der symbolischen Position zu erfüllen. Im Beispiel ist dies eine Menge Konfigurationen, in denen der Roboter Objekt A greift. Konfigurationen werden mittels Roadmap Planer verbunden. Es wird unterschieden zwischen „Roboter mit gegriffenem Objekt" (Transport) und „Roboter ohne gegriffenem Objekt" (Transfer). Um die geometrische

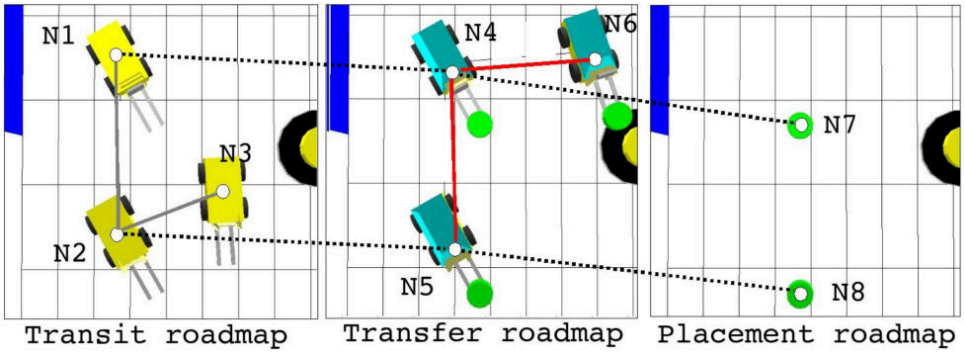


Abb. 3.4.: Positionen im aSyMov Planer. Die gepunkteten Linien zeigen die Assozierung der Knoten zwischen den verschiedenen Roadmaps. N1, N4 und N7 entsprechen z.B. der selben Position. Graue bzw. rote Kanten zeigen Übergänge in der Roadmap an. Quelle: (Cambon u. a., 2004)

Korrektheit zu gewährleisten, werden Roadmaps für unterschiedliche Roboter-/ Objektkombinationen erstellt. Ein Beispiel für einen Graphen, der aus Positionen aufgespannt wird, ist in Abb. 3.4 dargestellt.

Dabei ist die Herausforderung, dass eine sehr große Anzahl von Roboter-/ Objektkombinationen existiert. Für diese müssen potentiell Roadmaps erzeugt werden. Sie werden daher nach Bedarf des symbolischen Planers erstellt. Planung und Exploration neuer Roadmaps werden zyklisch ausgeführt; die Exploration wird von Heuristiken geleitet.

Der Planer benutzt ein dreidimensionales Modell seiner Umwelt. Als Roboter kommt ein sich auf einer zweidimensionalen Ebene bewegendes, simuliertes Fahrzeug zum Einsatz. Es gibt keine Manipulation. Griffe werden durch die Position des zweidimensionalen Roboters definiert. Das symbolische Modell ist vorgegeben, es gibt keine Anbindung an die echte Welt.

In (Dornhege u. a., 2010) wird ein System zur Integration symbolischer und geometrische Planung vorgestellt, das die Domänenmodellierungssprache PDDL (s. Abschnitt 2.1.4) um sogenannte „Semantische Anhänge" erweitert. Ein semantischer Anhang spezifiziert einen Aufruf eines externen Moduls zu einer Aktion. Dem Modul sind Zustandsmodell und Aktion sowie deren Parameter bekannt. Es bewertet aufgrund dieser Informationen, ob die Aktion im gegebenen Zustand ausgeführt werden kann. Konkret schlagen Dornhege et. al. die Anbindung eines Roadmap Bahnplaners vor. Dieser bestimmt für Bewegung- und Transportaktionen des Roboter, ob diese kollisionsfrei ausgeführt werden können.

Der symbolische Planer erzeugt regelmäßig redundante Anfragen an die semantischen Anhänge. In (Dornhege u. a., 2013) wird eine Cache Methode vorgestellt, die es vermeidet, diese Anfragen mehrmals an den geometrischen Planer weiterzugeben. Das System ermöglicht damit die Steuerung eines simulierten dreidimensionalen Roboters wie in Abb. 3.5. Es wird ebenfalls gezeigt, dass das System für Umgebungen mit anspruchsvollen Bahnplanungsaufgaben funktioniert.

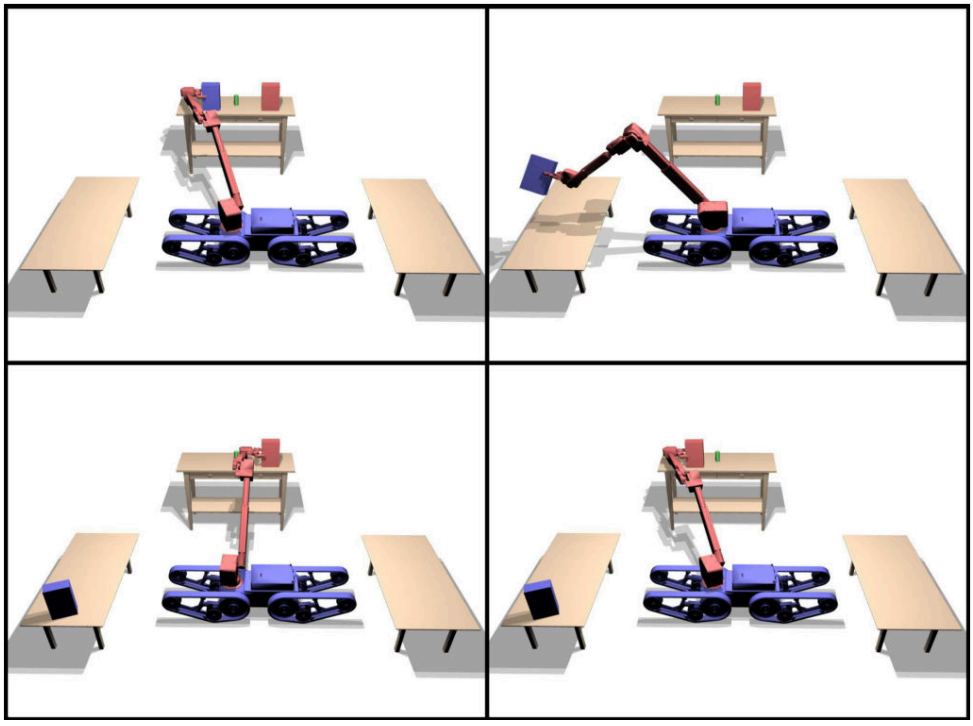


Abb. 3.5.: Erzeugter Plan aus (Dornhege u. a., 2010) in der simulierten Ausführung. Die rote Schachtel soll an die Stelle der blauen gelegt werden. Der Planer ist in der Lage, die Aktionssequenz zum Entfernen der blauen Box und Abstellen der roten unter Berücksichtigung der Kollisionsfreiheit zu planen. Quelle: (Dornhege u. a., 2010)

In (Dornhege u. Hertle, 2013) wird gezeigt, dass das System (mit Erweiterungen) auch in der Lage ist, einen Roboter in einer realen Umgebung zu steuern. Dazu wurde es um eine kontinuierliche Planungsschleife und eine Ausführungsüberwachung erweitert. Der Roboter kann mit dem System in der Umgebung Objekte zwischen unterschiedlichen Orten transportieren und eine *Wisch*-Aktion ausführen bzw. die Umgebung anpassen,um die Ausführung zu ermöglichen. Besonders hervorzuheben ist die Fähigkeit des Systems, mit unbekannten Eigenschaften der Szene umzugehen (Nebel u. a., 2013). Damit ist es dem System möglich, ein Objekt in der Umgebung zu suchen und dessen Ort aus Sensordaten zu gewinnen.

Das System beschreibt einen Ansatz zur Nutzung eines symbolischen Planers zur Robotersteuerung. Es lässt aber vor allem die Frage offen, wie über Trajektorienplanung und Positionsgenerierung von Objekten Wissen aus der realen Welt bzw. ihrem geometrischen Modell ins Symbolische transferiert werden kann. Beispielsweise ist unklar, wie eine Ablagepose eines aus dem Weg geräumten Objektes erzeugt wird. Außer Positionen können keine symbolischen Eigenschaften der Szene oder von Aktionen erzeugt werden. In den evaluierten Szenarien lagen die Planungszeiten im Bereich von 1 Minute für ein Objekt bis zu 11 Minuten für 5 Objekte. Die Online-Fähigkeit ist also nur eingeschränkt gegeben.

Ein System, das einen Roboter zum Planen von Handlungen als Küchenhelfer befähigt, wird in (Gravot u. a., 2006) vorgestellt. Es verwendet einen HTN Planer (s. Abschnitt 2.1.2), der an eine Datenbank mit Rezepten angebunden ist. Erwähnenswert ist die Fähigkeit des Systems, Aufgaben, zu deren Ausführung es selbst nicht in der Lage ist, an den Menschen zu delegieren, etwa das Abschmecken von Speisen. Die konkrete Anbindung der Manipulation ist auf zwei Fähigkeiten beschränkt: Zum Einen das Planen kollisionsfreier Trajektorien mittels RRTs zum Umsetzen von Transportaktionen. Zum Anderen die Ausführung vorgegebener Trajektoren zur Umsetzung spezifischer Aktionen. Weder auf Perzeptions- noch auf Ausführungsseite ist eine Anbindung an die Wirklichkeit umgesetzt. Wissen muss von einem Experten modelliert werden. Eine Adaption an neue Szenen ist mit dem Ansatz nur in engen Grenzen möglich.

Einen ähnlichen, jedoch weitergehenden Ansatz beschreibt (Karlsson u. a., 2012). Ein RRT Planer wird darin aus einem HTN Planer aufgerufen. Es wird ein Sampling im Geometrischen verwendet, um z.B. Ablageposen zu erzeugen. Besonders hervorgehoben wird, dass dieses geometrische Sampling ein Backtracking verwendet. Entscheidungen, die rein im Geometrischen erzeugt werden, können damit aus dem Symbolischen zurückgezogen werden, wenn sie keine Lösung erlauben. Die Abtastungsdichte des geometrischen Raumes ist in dem Ansatz eine kritische Größe. Ist sie hoch, verbringt der Planer viel Zeit in der geometrischen Planung. Ist sie zu niedrig, werden Lösungen nicht gefunden. Der Ansatz wurde auf dem Roboter Justin (Ott u. a., 2006) des DLR evaluiert. Es werden Objektpositionen aus dreidimensionalen Punktwolken wahrgenommen. Die Verbindung zwischen symbolischem und geometrischem Modell erfolgt ausschließlich über Anfragen, es wird kein Wissen in das symbolische Modell transferiert.

(Guitton u. Farges, 2009) erweitert die Domäne eines symbolischen Planer um *geometrische Vorbedingungen*. Diese können etwa besagen, dass ein Roboter in der Nähe eines Objektes sein oder eine bestimmte Ausrichtung haben muss. Plant der symbolische Planer eine Aktion, so wird diese mit ihren geometrische Vorbedingungen an den geometrischen Planer gegeben, der aus den Vorbedingungen eine konkrete geometrische Position für den Roboter erzeugt und im Anschluss einen Pfad von der aktuellen zur neuen Position erzeugt. Das Verfahren betrachtet auf geometrischer Ebene lediglich die Pfadplanung, es wird auf simulierten zweidimensionalen mobilen Robotern evaluiert. Auf symbolischer Ebene können auch weitere Aktionen geplant werden, wie etwa ein Foto aufzunehmen. Ihre geometrischen Vorbedingungen müssen von Hand modelliert werden.

Einen streng hierarchischen Planer präsentieren (Kaelbling u. Lozano-Perez, 2010). Es werden im Symbolischen HTNs verwendet, deren Blätter durch geometrische Planung ins Kontinuierliche abgebildet werden. Eine Besonderheit des Systems ist, dass bei jedem Aufsteigen aus dem HTN die geplanten Aktionen sofort ausgeführt werden. Dies wird „Planung in der Gegenwart"[6]

[6] engl.: planning in the now

genannt. Es wird nicht erst ein vollständiger Baum aufgebaut. Das hat zur Folge, dass der Folgezustand einer Aktion aus deren Ausführung bestimmt werden kann, es wird kein Prädiktionsfehler im symbolischen Modell erzeugt. Allerdings kann sich im späteren Verlauf der Planung herausstellen, dass der erstellte Plan nicht zielführend ist. In diesem Fall müssen die Aktionen rückgängig gemacht werden. Die Autoren gehen davon aus, dass dies meist möglich ist. Zur Integration der geometrischen Planer wird vorgeschlagen, vereinfachte Planer, die schnell auf Anfragen antworten können, zu verwenden. Dies ist in der betrachteten Aufgabe der simulierten mobilen Manipulation in einer zweidimensionalen Szene, z.B. durch Vernachlässigen der Roboterkinematik bei der Bahnplanung möglich. Wie weit diese Annahme auf Manipulation im dreidimensionalen Raum übertragbar ist, ist offen. Ein weiterer Aspekt der Arbeit sind die sogenannten *geometrischen Hinweisgeber*[7]. Sie ermöglichen es dem symbolischen Planer, bei Bedarf neue Symbole aus dem Kontinuierlichen zu erzeugen. Diese können etwa Positionen, Griffe oder Pfade beschreiben.

Der Ansatz von (Choi u. Amir, 2009) verwendet eine symbolische Planung von Manipulationsaktionen. Er wurde bereits in Abschnitt 3.1.3 vorgestellt.

3.2.2. Aufgabenspezifische Erweiterung von Bewegungsplanern

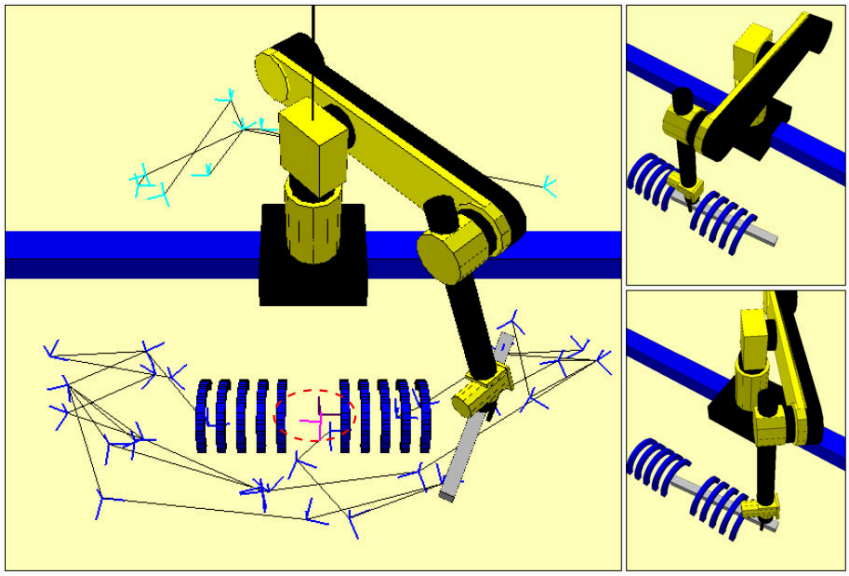


Abb. 3.6.: Beispielproblem, dass mittels modifizierter PRM Planung gelöst werden kann. Der Roboter verschiebt den Quader aus dem Käfig (oben rechts), sodass er zum Rand des Quaders umgreifen kann (unten rechts). Aus dieser Konfiguration kann der Roboter den Quader aus dem Käfig ziehen. Verwendet Roadmaps sind links dargestellt. Quelle (Simeon u. a., 2002)

[7] engl.: geometric suggester

Basierend auf ähnlichen Konzepten, wie sie auch dem im vorigen Abschnitt beschriebenen aSyMov Planer zugrunde liegen, schlagen (Simeon u. a., 2002) einen PRM basierten Manipulationsplaner vor, der in der Lage ist, Umgreifoperationen zu erzeugen. Ein Problem, das mit dem Ansatz lösbar ist, ist in Abb. 3.6 dargestellt. Es werden Roadmaps für den Arm generiert, sowie für einen Arm mit gegriffenem Objekt, wobei unterschiedliche Möglichkeiten berücksichtigt werden, das Objekt zu greifen. Verbindungen zwischen den Roadmaps bestehen an Positionen, an denen das Objekt stabil abgelegt werden kann. Damit ist ein PRM-artiger Graph definiert, auf dem die Planung als Suche durchgeführt werden kann. Griffe und Ablageposen werden aus dem Kontinuierlichen erzeugt. (Simeon u. a., 2004) evaluieren den Ansatz für einen simulierten mobilen Manipulator. Das Verfahren ist auf die Bewegung eines gegebenen Objektes ausgerichtet. Szenenwissen über andere Objekte wird nicht erzeugt; sie werden als unbewegliche Hindernisse betrachtet.

(Berenson u. a., 2007) integrieren einen Greif– und einen RRT-Pfadplaner zu einem Pick–and–Place Planer. Dazu berechnen sie offline eine Menge von Griffen für unterschiedliche Objekte. Insbesondere enthält die erstellte Griffdatenbank für jeden Griff eine zu erreichende TCP-Anrückpose und Bewertungsmaße zur Auswahl der Griffe. Online können so die Start- und Zielposen der Objekte auf TCP Posen abgebildet werden. Damit ein Griff zur Anwendung kommen kann, müssen Start- und Zielpose kinematisch erreichbar sein. Ist dies der Fall, plant ein RRT-Bahnplaner eine Trajektorie zwischen den Start– und Zielkonfigurationen. Ist dies nicht möglich, werden sukzessive weitere Griffe ausgewählt. In (Berenson u. a., 2008) wird das Verfahren für mobile Manipulation erweitert, dabei ist vor allem die Roboterkonfiguration zum Greifen relevant, da diese auch für die mobile Plattform erzeugt werden muss. Um sie zu erzeugen wird ein Optimierungsverfahren eingesetzt. In (Berenson u. a., 2011) wird das Verfahren um die *Aufgabenraum-Region*[8] erweitert. Sie erlaubt es, zu erreichende Posen nicht nur exakt, sondern im Rahmen von begrenzten Bereichen zu definieren.

In (Stilman u. a., 2006) wird ein Planer für das Problem der *Navigation unter beweglichen Objekten* vorgestellt. Dieser erzeugt eine Aktionssequenz für einen mobilen Roboter, der in der Lage ist, Objekte zu verschieben bzw. zu tragen. Es werden Aktionen generiert, die es dem Roboter erlauben, einen Weg frei zu räumen. Dabei wird der Raum in zusammenhängende Partitionen segmentiert, in denen der Roboter navigieren kann, ohne Objekte verschieben zu müssen. Diese Partitionen werden durch Verschiebeaktionen verbunden. Die Planung erfolgt in einer zweidimensionalen Projektion der Umgebung. Unterhalb des globalen Planers werden weitere Planer verwendet, die die Bewegungen zum Verschieben der Objekte und die Trittpunkte für einen zweibeinigen Laufroboter zur Lokomotion planen. Das Verfahren wurde auf einem HRP 2 evaluiert.

[8] engl.: task space region

Abb. 3.7.: Planungsszenario des Manipulationsplaners aus (Stilman u. a., 2007). Der Hammer soll gegriffen werden, dazu müssen die beiden Objekte davor entfernt werden. Sie werden an durch Abtastung erzeugt Orte gelegt. Angepasst übernommen aus (Stilman u. a., 2007).

Ein Verfahren zur Bahnplanung zwischen beweglichen Objekten wird in (Stilman u. a., 2007; Schamburek, 2006) präsentiert. Dabei wird zuerst in einem hindernisfreien Raum mit einem RRT eine Transportbewegung für ein Objekt bestimmt. Das Verfahren ist in der Lage, dabei unterschiedliche Griffe zu berücksichtigen. Aus der erzeugten Trajektorie wird der benötigte Freiraum bestimmt; Objekte, die sich in diesem befinden sind Hindernisse, die entfernt werden müssen. In einem Sampling-Schritt werden mögliche Posen zum Ablegen der Hindernisse erzeugt, die sich auf Ebenen befinden und nicht mit bereits erzeugten Pfaden kollidieren. Rekursiv wird das Verfahren dann auf die Hindernisse angewandt.

Das Verfahren nimmt an, dass das Ergebnis ein monotoner Plan ist, d. h. dass jedes Objekt nur einmal bewegt wird. Damit ist die Länge des resultierenden Planes begrenzt. Allerdings dürfen die Bewegungen der Hindernisobjekte nicht dazu führen, dass ein Objekt bewegt wird, dass bereits in der Menge der Hindernisobjekte ist. Um dies zu verhindern, wird die Reihenfolge, in der Hindernisobjekte entfernt werden, variiert. Ein Szenario, in dem der Planer evaluiert wurde, ist in Abb. 3.7 abgebildet. Es wurden ebenfalls Szenarien betrachtet, in denen die kinematisch stark einschränkende Bewegung der Schranktür ausgeführt wird. Das Verfahren modelliert die Abhängigkeiten der Aktionen rein in geometrischen Modellen, eine Erweiterung mit symbolischem Wissen ist nicht möglich. Insbesondere kann kein mechanisches Szenenwissen dargestellt werden.

In (Jäkel u. a., 2010; Jäkel, 2013) wird ein Verfahren vorgestellt, das gelernte Aktionen an eine Szene anpasst. Dazu wird ein Mensch bei der Ausführung einer Aktion beobachtet. Das Verfahren fällt also auch in den Bereich des Programmierens durch Vormachen, der Anteil des Lernen an dem Verfahrens ist jedoch für die Analyse aus der Perspektive dieser Arbeit nicht relevant. Aus der Beobachtung wird ein *Strategie-Graph* erzeugt. Dieser enthält in jedem Knoten

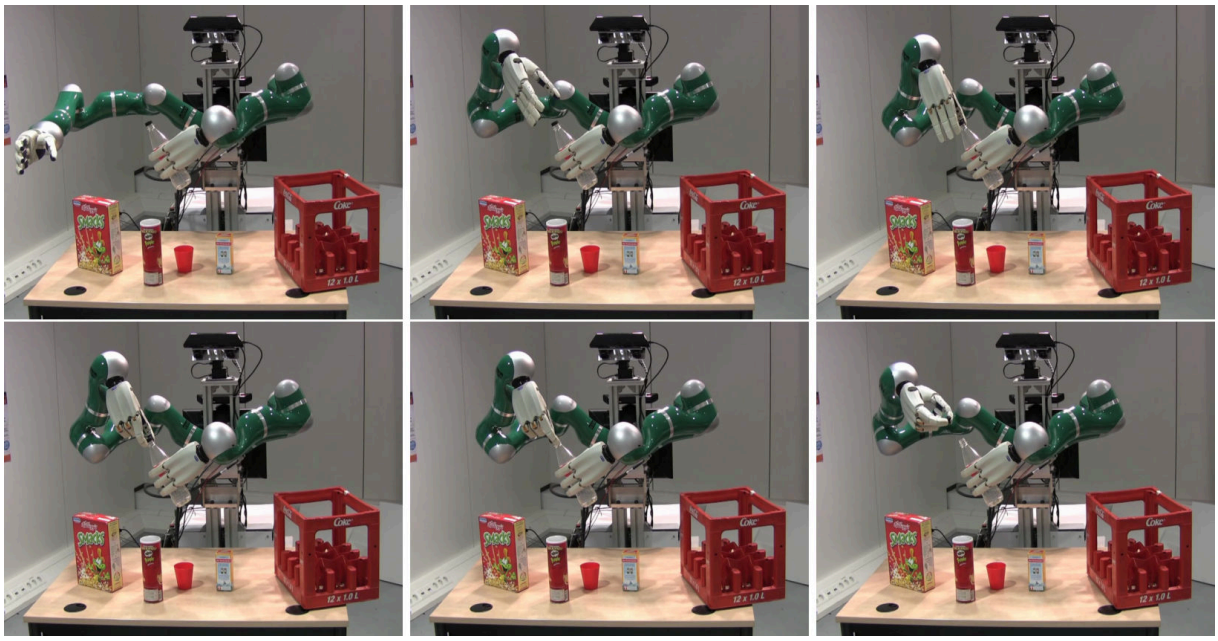


Abb. 3.8.: Geplante Ausführung einer „Flasche öffnen" Aktion. Die Aktion ist aus mehreren Teilaktionen, die in einem Strategie-Graphen modelliert sind, zusammengesetzt. Die Finger- und Handpositionen ergeben sich aus Bedingungen und werden abhängig von Szene und Roboter geplant. Aus (Jäkel u. a., 2012)

Bedingungen[9], die relevante Aspekte der Aktion beschreiben. Diese können beispielsweise die Position der Fingerspitzen relativ zum Objekt sein, oder die Höhe eines gegriffenen Objektes über dem Tisch. Unterschiedliche Knoten im Strategie-Graph enthalten unterschiedliche Bedingungen, sodass andere Konfigurationen des Roboters bzw. der Objekte erzwungen werden. Es wird ein RRT Planer verwendet, der im Raum der durch die Bedingungen erlaubten Konfigurationen nach Pfaden sucht, welche die Aktion ausführen. Ein Beispiel für eine geplante Ausführung ist in Abb. 3.8 zu sehen.

Das Verfahren ist in der Lage, auch Aktionen, die im Sinne dieser Arbeit als Aktionssequenzen aufgefasst werden, zu planen. Z. Bsp. wird in einem Beispiel eine Flasche gegriffen und in einem Kühlschrank abgestellt. Dabei ist jedoch die Anzahl und Art der Aktionen fest, es können keine neuen Aktionen erzeugt werden. Auch gibt es kein Modell, wie sich die Aktionen des Roboters auf die Szene auswirken.

(Kresse u. Beetz, 2012) beschreiben einen Ansatz, um symbolisch bestimmte Randbedingungen wie „zeigt auf", „bewege unter" oder „nahe" auf Randbedingungen für Roboterkonfigurationen abzubilden. Diese werden in einem Bewegungsplaner als Zielkonfigurationen eingesetzt. Versuche in der Simulation zeigen, dass das Verfahren für mehr Konfigurationen valide Lösungen findet, als der Ansatz über starre Referenz-Frames. Neue Aktionen werden jedoch nicht erzeugt.

[9] engl.: constraints

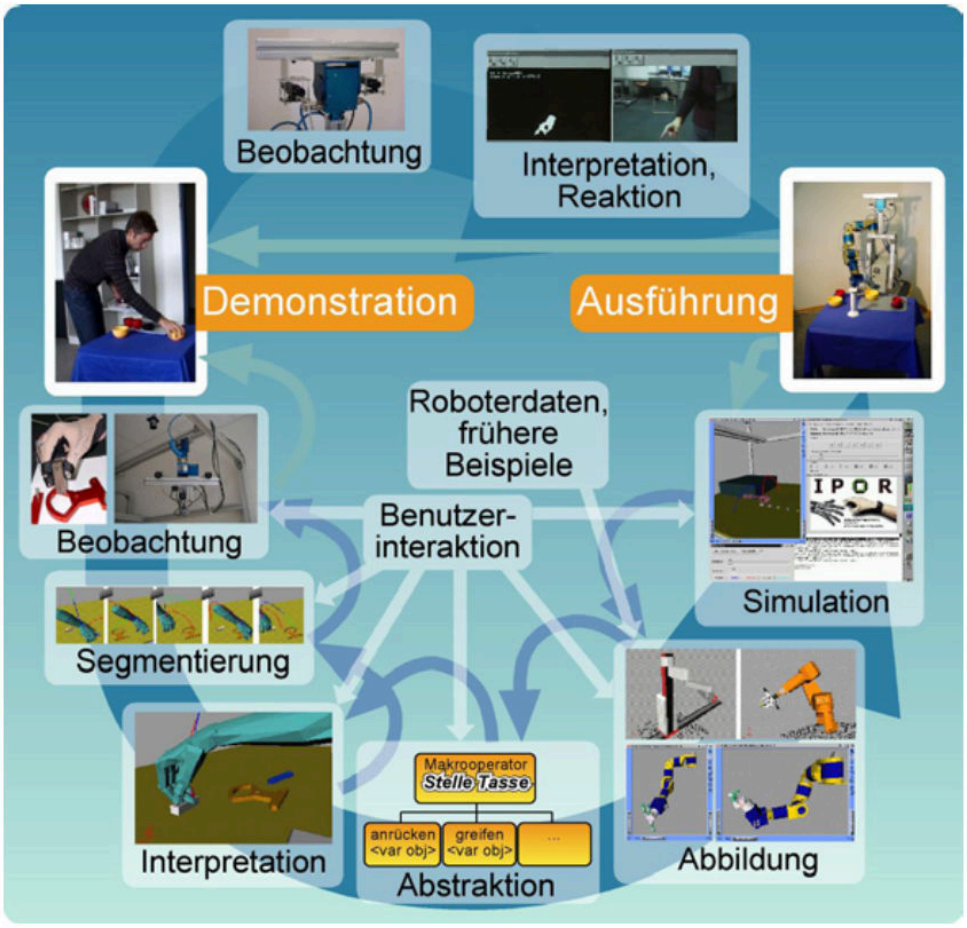


Abb. 3.9.: PDV Zyklus nach (Rogalla, 2003). Beginnend bei der Demonstration des Benutzers werden die Schritte der Beobachtung, Segmentierung, Interpretation und Abstraktion durchgeführt. Das Ergebnis wird auf einen konkreten Roboter abgebildet und vor der Ausführung in einer Simulation verifiziert. Die Schritte erfolgen unter Einbeziehung des Benutzers, es gibt keine Adaption des Gelernten an eine wahrgenommene Szene.

3.2.3. Programmieren durch Vormachen

Der Ansatz des *Programmierens durch Vormachen* (PdV) hat zum Ziel, Wissen über Handlungen aus Benutzerdemonstrationen zu erzeugen. Es lassen sich folgende Forschungsrichtungen unterscheiden (Zöllner, 2006):

- Lernen von Elementarfähigkeiten: Einzelne Fähigkeiten[10] werden in enger Kopplung von Sensoren und Aktoren direkt am Roboter gelernt
- Imitationslernen[11] Einzelne Fähigkeiten werden aus Beobachtung des Menschen gelernt
- Handlungslernen: Methoden zum Erlernen komplexer Handlungsfolgen mit meist symbolischer Wissensrepräsentation

[10] engl.: skills
[11] engl.: imitation learning

Nur das Handlungslernen beschäftigt sich mit Aktionsfolgen und ist damit für diese Arbeit relevant. Eine Reihe von Arbeiten lassen sich unter diesem Aspekt betrachten: (Friedrich, 1999; Rogalla, 2003; Ehrenmann, 2003; Zöllner, 2006; Knoop, 2007; Pardowitz, 2007; Lösch, 2012; Schmidt-Rohr, 2013).

Einen generelle Unterteilung des Prozesses des PdV wird in (Rogalla, 2003) beschrieben. Sie ist in Abb. 3.9 dargestellt. Es existieren die folgenden Schritte, wobei nicht jede Arbeit alle Schritte behandelt:

Beobachtung Sie bezeichnet die Aufzeichnung einer Benutzerdemonstration zu Serien von Sensorwerten. Dabei kommen abhängig von der beobachteten Handlung unterschiedliche Sensoren zum Einsatz, z.B. Kameras und 3D Sensoren (Lösch, 2012), Exo-Skelette oder Datenhandschuhe zur Beobachtung der Hände und Finger (Zöllner, 2006). Neben den Aktionen des Benutzers sind auch die Veränderungen an Objekten von Interesse.

Segmentierung Die Segmentierung unterteilt den kontinuierlichen Datenstrom aus der Beobachtung in Segmente, denen eine Semantik innerhalb des Segmentes bezüglich der Handlung gemeinsam ist. Für diesen Schritt wird zusätzlich ein Handlungsmodell benötigt.

Interpretation Im Interpretationsschritt wird die beobachtete Handlung mit ihren Auswirkungen auf die Umwelt in Verbindung gebracht. Es wird bestimmt, was (beabsichtigte) Effekte der Handlung sind.

Abstraktion Im Abstraktionsschritt wird eine generalisierte Beschreibung der gelernten Handlung erzeugt. Sie erlaubt es, in den späteren Schritten das Wissen auf die Anwendung zu übertragen.

Abbildung In diesem Schritt wird die Handlung auf ein Robotersystem übertragen. Dabei werden symbolischen Aktionsbeschreibungen konkrete, für den Roboter ausführbare Programme zugeordnet. Diese können z.B. auf der Ebene von Trajektorien definiert sein.

Simulation Das erzeugte Programm wird in einer simulierten Umgebung, z.B. einer graphischen Visualisierung, ausgeführt. Dabei kann überprüft werden, ob die simulierten Effekte den Erwartungen in der Anwendung entsprechen. Werden Fehler entdeckt, so muss im Prozess zurückgegangen werden, um diese zu korrigieren.

Ausführung Im letzten Schritt des Prozesses kommt das gelernte Programm auf dem Roboter zur Ausführung.

In den betrachteten Arbeiten ist der Mensch in die Schritte des PdV eng eingebunden. Auch ist grundsätzlich die Wahrnehmung der Szene nur in der Ausführungskomponente vorgesehen, die anderen Prozesschritte sind daran nicht angebunden. Die Symbolverankerung wird in den Ansätzen nicht tief gehend behandelt, entsprechen fehlt die Möglichkeit zu szenenabhängigen

Adaption. Der Aspekt der Benutzerbeobachtung ist in dieser Arbeit von untergeordneter Bedeutung. Im Folgenden werden kurz auf die Beiträge unterschiedlicher Arbeiten zur Handlungsgenerierung eingegangen.

Zur symbolischen Beschreibung von Handlungen werden beim PdV Operatoren verwendet. Nach (Friedrich, 1999) wird unterschieden zwischen Elementaroperatoren, die direkt ausführbar sind und Makrooperatoren, welche in weitere Operatoren zerlegt werden können. (Knoop, 2007) beschreibt die Abbildung gelernten Wissens auf die Ausführung eines Roboters. Dazu führt er die Flexiblen Programme (s. Abschnitt 2.1.3) ein, die aus gelernten Makrooperatoren erzeugt werden. Die Erzeugung erfolgt interaktiv. Ein Experte betrachtet die Visualisierung des Simulationsschrittes und erkennt Fehler in der Ausführung. Er passt darauf die Parametrierung der vorhergehenden Schritte an, um das Ergebnis zu verbessern. Die Szene, auf der das Programm ausgeführt wird, wird nicht berücksichtigt.

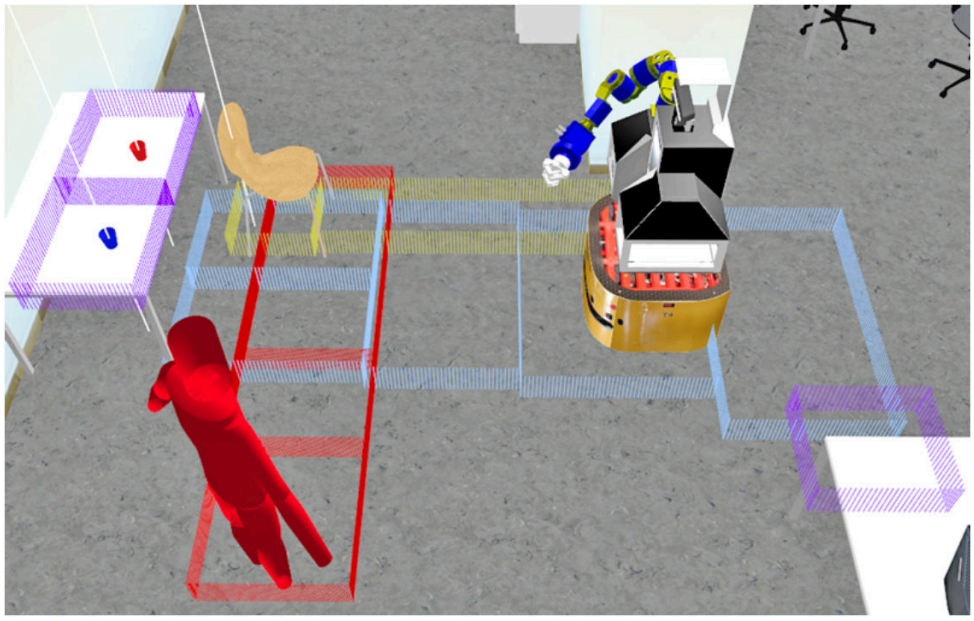


Abb. 3.10.: Symbolisierung der Positionen unterschiedlicher Entitäten in einer Szene. Ein Feld entspricht einem Symbol. Rot dargestellt sind Regionen, in denen sich der Benutzer aufhalten kann, blau Roboterpositionen, gelb Positionen des Stuhls. Letztere Positionen sind auf den Boden, für Objekte werden die lila Regionen auf den Tischen modelliert. Quelle: (Schmidt-Rohr, 2013)

Einen PdV Ansatz zum Lernen von Entscheidungsstrategien für Manipulationshandlungen stellt (Schmidt-Rohr, 2013) vor. Entscheidungen werden auf Missionsebene getroffen, d. h. es wird Benutzerinteraktion, Mobilität und Manipulation berücksichtigt. Es wird ein Modell in einem partiell beobachtbaren Markov Entscheidungsprozess erzeugt (POMDP). Dieses basiert auf einem symbolischen Zustandsmodell, das aus Belehrungsvorgängen erzeugt wird. Dazu werden die Entitäten der Demonstration durch Merkmalsfunktionen auf kontinuierliche Vektoren abgebildet, etwa die Benutzerposition. Durch Clustering-Methoden werden diese zu Symbolen zu-

sammengefasst. Ein Beispiel ist in Abb. 3.10 gezeigt. Zur Ausführung können die Entitäten über die Merkmalsfunktion der erzeugten Symbole, welche ihre Position repräsentieren, zugeordnet werden. Zur Modellierung weiteren Wissens über die Szene werden Ontologien eingesetzt. Aktionen werden aus den Übergängen zwischen den Zuständen erzeugt, ebenso gibt es vordefinierte Aktionen. Aus dem POMDP werden Entscheidungsregeln erzeugt, die Zustände auf auszuführende Aktionen abbilden. Sie maximiert eine Belohnungsfunktion. Die Möglichkeit des Verfahrens, auf neue Umgebungen zu generalisieren, ist stark abhängig von den gelernten Umgebungen. Für Manipulationsaktionen würde eine sehr feine Diskretisierung der Regionen der Entitäten benötigt, mit der der Zeitaufand für das Lernverfahren inakzeptabel hoch wäre. Innerhalb der verwendeten Einsatzszenarien ist die Methode echtzeitfähig und in der Lage, sich begrenzt veränderten Szenen anzupassen. Auch zusätzliche Aktionen in einer Sequenz können erzeugt werden.

Auch das Verfahren von (Jäkel, 2013) nutzt Elemente des PdV. Es wurde in Abschnitt 3.2.2 unter seinen Aspekten als Bewegungsplaner beschrieben. Es ist für die Generierung neuer Handlungen aber nicht in entsprechendem Maße geeignet.

3.2.4. Fazit

Im Bereich der Handlungsgenerierung lässt sich ein breiter Stand der Technik feststellen. In dieser Analyse wurde daher explizit nur auf den Bereich der Planung mit Bezug zur Robotik eingegangen, reine KI Ansätze im Stile der Blocksword Planung bleiben außen vor. Eine große Gruppe an Verfahren verbinden symbolische mit kontinuierlichen Planern, indem zu erzeugten Symbolen Aufrufe an die kontinuierlichen Planer erzeugt werden. Dabei werden aus dem Symbol zwei Punkte im geometrischen abgeleitet und ein kontinuierlicher Planer sucht nach einem verbindenden, kollisionsfreien Pfad (Gravot u. a., 2005; Dornhege u. a., 2010; Gravot u. a., 2006; Karlsson u. a., 2012; Guitton u. Farges, 2009). Einige Verfahren berücksichtigen beim Erzeugen der Punkte die Greifplanung. Lediglich (Gravot u. a., 2006), (Dornhege u. Hertle, 2013) und (Karlsson u. a., 2012) planen im dreidimensionalen Raum, die beiden letzteren evaluieren auch auf einem echten Roboter. Das Verfahren von (Choi u. Amir, 2009) verwendet eine umgekehrte Reihenfolge im Vergleich zu den vorherigen Arbeiten und erzeugt in einem Vorverarbeitungsschritt Aktionssymbole mit einem kontinuierlichen Planer. Auch dort wird nur Bahnplanung benutzt. Weder bilden die Verfahren Aktionswissen für neue Aktionen in den symbolischen Planungsraum ab, noch gibt es einen integrierten Ansatz, der Szenenwissen, wie in Abschnitt 3.1.2 beschrieben, berücksichtigt.

Eine weitere Klasse von Verfahren verzichtet auf den symbolischen Planer und erweitert Roadmap basierte Ansätze. Dazu werden zusätzliche Samples in der Roadmap erzeugt, von denen bekannt ist, dass an diesen ein Zustandsübergang, der eine Aktion einleitet, möglich ist. Sol-

Verfahren	Anwendung	Manipulation	Neue Aktionen	Wissenstransfer	Adaption	Online	Wirklichkeit	Evaluiert
aSyMov (Gravot u. a., 2005)	Simulierte mobile Roboter	⊖	⊕	⊗	⊕	?	⊗	⊗
Planung mit semant. Anhängen (Dornhege u. Hertle, 2013)	Mobile Manipulation	⊕	⊕	⊖	⊕	⊖	⊕	⊕
RRT aus HTN (Karlsson u. a., 2012)	Manipulation	⊕	⊕	⊖	⊘	⊖	⊕	⊕
Planung in der Gegenwart (Kaelbling u. Lozano-Perez, 2010)	Mobile Manipulation	⊗	⊕	⊗	⊕	⊕	⊗	⊗
Planung mit gesampelten Aktionen (Choi u. Amir, 2009)	Manipulation, Pick and Place	⊕	⊕	⊗	⊖	⊖	⊗	⊗
RPM mit Greifen (Simeon u. a., 2002)	Manipulation, Pick and Place	⊕	⊘	⊗	⊖	⊖	⊗	⊗
RRT mit Greifen (Berenson u. a., 2007)	Manipulation, Pick and Place	⊕	⊗	⊗	⊖	⊖	⊕	⊕
Manipulation u. beweglichen Objekten (Stilman u. a., 2006)	Manipulation, Pick and Place	⊕	⊕	⊖	⊕	⊖	⊘	⊖
Planung von Manipulations-strategien (Jäkel, 2013)	Manipulation	⊕	⊗	⊗	⊘	⊕	⊕	⊕
Programmieren durch Vormachen	Servicerobotik	⊕	⊘	⊘	⊖	⊗	⊘	⊕
POMDP PdV (Schmidt-Rohr, 2013)	Servicerobotik	⊕	⊕	⊘	⊘	⊕	⊕	⊕

Tab. 3.2.: Überblick über die diskutierten Verfahren im Bereich der Handlungsgenerierung. ⊕: gut geeignet; ⊘ mit Einschränkungen geeignet, ⊖ schlecht geeignet, ⊗ ungeeignet, ?: unbekannt, leeres Feld: Kriterium kann nicht angewandt werden. Zur Bedeutung der Kriterien, vlg. Anfang des Kapitels.

che Samples können etwa durch stabile Objektpositionen definiert sein, die das Greifen oder Abstellen eines Objektes ermöglichen (Simeon u. a., 2002; Berenson u. a., 2007; Stilman u. a., 2006; Schamburek, 2006). Die Verfahren fokussieren dabei stark auf die Bewegungsgenerierung. (Schamburek, 2006) nutzt Heuristiken aus Szenenwissen, um sinnvolle Aktionsreihenfolgen zu erzeugen. Die fehlende symbolische Ebene macht die Modellierung von Szenenwissen jedoch schwierig. Auch ist keines der Verfahren performant genug, um online eingesetzt werden zu können.

Das Programmieren durch Vormachen sticht vor allem durch seine Lernfähigkeit neuer Aktionen aus Benutzerdemonstrationen hervor. Bei den betrachteten Arbeiten wurde jedoch die Ausführung in unterschiedlichen Szenen kaum betrachtet. Wenn überhaupt sind nur kleine Varianzen erlaubt; ein allgemeines Modell der Szene und die Anpassung der Handlungssequenz

darauf wurde nicht gefunden. Gerade hier knüpft diese Arbeit an, einen zu Beitrag liefern, der auch zur Erweiterung von PdV Methoden um eine szenenabhängige Anpassung genutzt werden kann.

3.3. Ausführungsüberwachung

> Ausführungsüberwachung ist die kontinuierliche Echtzeitaufgabe, durch Aufnehmen von Informationen und Erkennen von Anomalien des Verhaltens die Bedingungen eines physikalischen Systems zu bestimmen [12]. (Übersetzung der Definition aus (Isermann, 2004))

Ausführungsüberwachung ist keine robotikspezifische Aufgabenstellung, sondern kommt beim Ausführen verschiedenster Prozesse zum Einsatz. In der Steuerungs- und Regelungstechnik wird sie meist als *Fehlererkennung und Isolation* bezeichnet[13](Pettersson, 2005). *Fehlererkennung* stellt fest, dass im System ein Fehler existiert, *Fehlerisolation* klassifiziert, wo der Fehler ist und die *Fehleridentifikation* bestimmt die Größenordnung des Fehlers. Im Folgenden werden Arbeiten mit Bezug zur Robotik untersucht.

(Pettersson, 2005) klassifiziert Ausführungsüberwachung in *analytische*, *datengetriebene* und *wissensbasierte* Ansätze. Die Unterteilung wird in dieser Arbeit übernommen. Analytische Ansätze benutzen mathematische oder physikalische Modelle, mit denen die Ein- und Ausgabe eines Systems auf je einen beschreibenden Wert abgebildet werden. Aus der Differenz der Werte wird ein Residuum gebildet. Ist dieses null bzw. betragsmäßig kleiner als ein Schwellwert, so ist der Systemzustand fehlerfrei. Modellbasierte analytische Ausführungsüberwachung ist in der Robotik der verbreitetste Ansatz (Pettersson u. a., 2005). Die datengetriebenen Methoden verwenden kein Modell des Systems, sondern arbeiten direkt auf gemessenen Daten. Diese werden beispielsweise mit statistischen Methoden klassifiziert. Wissensbasierte Methoden verwenden das Vorgehen eines menschlichen Experten, um Fehler zu erkennen. Im Folgenden wird eine Auswahl von für diese Arbeit relevanten Verfahren vorgestellt. Diese wurden im Bereich der analytischen und wissensbasierten Ansätze identifiziert. Im Bereich der datengetriebenen Ansätze ist kein für diese Arbeit relevanter Beitrag bekannt. In (Pettersson, 2005) finden sich zahlreiche weitere Verfahren.

Analytische Ansäzte Zahlreich analytische Ansätze zu Erkennung von Kollisionen während der Ausführung existieren: (de Luca u. Mattone, 2005) nutzen das dynamische Modell

[12] engl.: Execution monitoring is a continuous real-time task of determining the conditions of a physical system, by recording information, recognizing and indicating anomalies in the behavior.

[13] engl.: fault detection and isolation

eines Manipulators, um Kollisionen rein aus den Daten der Gelenkwinkel-Encoder zu detektieren. Daraus können sie das kollidierende Glied der kinematischen Kette bestimmen und eine auftretende Kraft abschätzen. Ebenfalls zur Kollisionserkennung nutzen (Morinaga u. Kosuge, 2003) sowie (Luca u. a., 2006) die Impedanz-Regelung eines Manipulators. (Kawabata u. a., 2002) schlägt eine Kollisionserkennung aufgrund der Analyse von Motorströmen vor.

Ein weiterer Bereich, in dem analytische Ansätze zum Einsatz kommen, ist die Fehlerzustandserkennung bei mobilen Robotern: In (Dearden u. a., 2004) werden unterschiedliche Partikelfilter zur Zustandsschätzung für einen mobilen Rover eingesetzt. Die Zustandsbeschreibung enthält ein Fehlermodell mit entsprechender Zustandsprädiktion und schätzt darüber die Wahrscheinlichkeit eines Fehlerzustandes. (Gat u. a., 1990) verwenden einen Simulator, um zulässige Wertebereiche von Sensoren in Abhängigkeit von anderen Sensorwerten zur Fehlererkennung für einen Rover zu erzeugen.

Wissensbasierte Ansätze (Pettersson u. a., 2005) verwenden ein neuronales Netz mit radialen Basisfunktionen zur Überwachung eines mobilen Roboters. Als Eingabe des Netzes dienen die Aktivierungsgrade der Verhalten eines definierten Zeitraums. Das Netz wird in einer Simulation trainiert. In einer Evaluation auf einem realen Roboter ist es in der Lage, Ausführungsfehler, die durch geschlossene Türen entstehen, zu erkennen.

Ausführungsüberwachung von Plänen (Ambros-Ingerson u. Steel, 1988) erstellen ein Modell eines Plans, das über Vor- und Nachbedingungen die Abhängigkeit von Aktionen voneinander modelliert. Werden diese verletzt, weil das Ergebnis einer Aktion anders ausfällt als erwartet, erfolgt eine Neuplanung. Das System ist auf rein logischer Ebene konzeptioniert und evaluiert.

(Hähnel u. a., 1998) schlägt mit *GOLEX* eine Sprache zur Beschreibung von Roboteraktionen vor. Diese bietet eine Möglichkeit, regelbasiert Erfolg und Misserfolg von Aktionen zu bestimmen. Die Regeln müssen, ebenso wie das Aktionswissen, von einem Experten erzeugt werden.

(Micalizio u. Torasso, 2007) schlagen ein System zur Überwachung eines Plans vor, der durch mehrere Agenten dezentral ausgeführt wird. Aktionen der einzelnen Agenten können dabei von den Aktionen anderer Agenten abhängen. Es wird ein Glaubensgrad über den Zustand der einzelnen Agenten aufgebaut, aus dem auf das Gesamtsystem geschlossen wird. Werden Abhängigkeiten verletzt, können Aufgaben neu zugeteilt werden, um die weitere Ausführung des Plans zu ermöglichen. Das Verfahren wurde auf mobilen Transportrobotern evaluiert.

Fazit Es konnten drei für die Robotik relevante Bereiche der Ausführungsüberwachung identifiziert werden: Im Bereich der Manipulation fokussieren bestehende Arbeiten auf die Kolli-

Verfahren	Anwendung	Manipulation	Neue Aktionen	Wissenstransfer	Adaption	Online	Wirklichkeit	Evaluiert
Analytische Kollisionserkennung	Manipulation	$\oplus$			$\oplus$		$\oplus$	$\oplus$
Analytische und wissensbasierte Bewegungsüberwachung	Mobile Robotik	$\otimes$			$\oplus$		$\oplus$	$\oplus$
Logische Überwachung von Aktionseffekten	Symbolische Planung	$\otimes$	$\oslash$	$\oplus$	$\oplus$		$\otimes$	$\ominus$

Tab. 3.3.: Überblick über die diskutierten Verfahren im Bereich der Ausführungsüberwachung. $\oplus$: gut geeignet; $\oslash$ mit Einschränkungen geeignet, $\ominus$ schlecht geeignet, $\otimes$ ungeeignet, leeres Feld: Kriterium kann nicht angewandt werden. Zur Bedeutung der Kriterien, vlg. Anfang des Kapitels.

sionserkennung. Dazu werden interne Sensoren der Manipulatoren modellbasiert ausgewertet. Im Bereich der mobilen Robotik wird auf ein generelles Fehlschlagen von Bewegungsprimitiven abgezielt; dies kann durch Kollisionen, Hardwaredefekte oder Festfahren bedingt sein. Das Ergebnis ist das Nicht-erreichen eines Zielortes, das frühzeitig erkannt werden soll. Die dritte Kategorie betrifft die Ausführung symbolischer Pläne. Dabei werden auf Basis logischer Modelle Vorbedingungen und Effekte identifiziert, die in gewissen Zuständen einen Fehlschlag des Plans zur Folge haben. Es bleibt eine Lücke bei der Ausführungsüberwachung von (absichtlich) kollisionsbehafteter Manipulation festzuhalten.

4. Ansatz zur szenenabhängigen Handlungsadaption

In diesem Kapitel wird untersucht, in welcher Form eine Adaption von Handlungssequenzen an eine Szene für einen Serviceroboter nützlich ist. Darauf basierend wird eine Problemanalyse durchgeführt, in der eine geeignete Methodik zur Adaption von Handlungssequenzen identifiziert wird. Es wird eine grundsätzliche Methodik zur Abstraktion der Wirklichkeit auf Modelle beschrieben, die es ermöglichen mit der Komplexität der Wirklichkeit umzugehen. Mit den Modellen können Vorhersagen über die Wirklichkeit getroffen werden, die letztlich Planung ermöglichen.

Dazu wird ein Metamodell entwickelt. Es beschriebt ein formales Modell für den Prozess der Adaption und Ausführung. Es werden Bereiche identifiziert, die neuartige Lösungen benötigen, um die Methodik anwenden zu können. Zum Abschluss des Kapitels wird ein System konzipiert, dass den Anforderungen der Problemstellung genügt und in dem die vorgeschlagenen Methoden umgesetzt werden.

4.1. Problemstellung

Um Handlungen an eine Szene zu adaptieren wird angenommen, dass es eine oder mehrere Aktionen gibt, die in einer Umgebung durch einen Roboter ausgeführt werden sollen. Eine solche Aktion kann das Greifen eines Objektes oder das Öffnen einer Flasche sein. Im einfachsten Fall existiert keine Abhängigkeit zwischen unbeteiligten Objekten der Szene und der auszuführenden Aktion, dann kann die Aktion direkt ausgeführt werden.

In real auftretenden Situationen ist das jedoch im Allgemeinen nicht gegeben. Dort existieren Objekte, die nicht direkt mit der Aktion zusammenhängen, diese jedoch als Hindernisse beeinflussen. Wie in Kapitel 2 und 3 dargelegt wurde, existieren Verfahren, die in der Lage sind, Aktionen anzupassen, sodass Kollisionen mit Hindernissen und damit deren Beeinflussung vermieden werden. Jedoch kann es Szenen geben, in denen diese Verfahren keine Lösung finden, etwa, weil Hindernisse zu dicht stehen und jeden möglichen Anrückpfad für den Roboter blockieren.

Auch entspricht das reine Anpassen der Aktion nicht dem Vorgehen des Menschen. Dieser bevorzugt es, in einer aufgeräumten Umgebung zu agieren und nimmt dazu zusätzliche Aktionen

in Kauf. Für den Menschen liegt die Motivation dafür auf der Hand: in einer aufgeräumten Umgebung haben Fehler bei der Bewegungsausführung weniger unerwünschte Konsequenzen, als in einer dicht mit Hindernis bestückten. Auch auf den Roboter ist diese Motivation übertragbar. Zwar können Industrieroboter Aktionen mit für die Kollisionsvermeidung vernachlässigbar geringen Abweichungen ausführen; für die Ausführung in der Robotik entstehen Fehler dagegen durch verrauschte Sensorik und unvollständige Modelle der Umgebung. Die Konsequenzen sind vergleichbar mit denen der ungenauen Ausführung des Menschen. Auch ein Roboter gewinnt bei der Ausführung in einer aufgeräumt strukturierten Umgebung an Sicherheit, bzw. er erhöht die Erfolgsaussicht seiner Handlung.

4.2. Problemanalyse

Um eine strukturierte Ausführungsumgebung zu erzeugen, müssen zusätzliche Aktionen geplant und ausgeführt werden, welche die Umgebung im Sinne der Strukturierung anpassen. In der angepassten strukturierten Umgebung können die Zielaktionen dann ausgeführt werden. Damit die erzeugten Aktionen das Ziel einer strukturierten Umgebung erreichen können, muss Wissen über Szene und Aktionen modelliert werden. Folgende Aspekte müssen berücksichtigt werden:

Wissen über Zielaktionen Es wird ein Modell der Zielaktion benötigt. Dieses bestimmt, was eine strukturierte Ausführungsumgebung für diese Aktion ist. Ein wesentlicher Teil diese Modells ist der benötigte Freiraum einer Aktion, in dem Kollisionen eine kontrollierte Ausführung verhindern würden. Weiter ist die Interaktion von manipulierten Objekten mit anderen Objekten von Interesse, etwa im Falle des Greifens von gestapelten Objekten. Aus dem Modell lassen sich Vorbedingungen ableiten, die zur Ausführung der Aktion erfüllt sein müssen.

Zielaktionen können in ihrer Ausführung von der Szene abhängen, etwa wenn sie von einem Manipulationsplaner aufgrund des Freiraumes erzeugt werden. Hier können auch flexible Planer zum Einsatz kommen, die automatisch Aktionen mit unterschiedlicher Semantik planen. Als Beispiel sei dazu das Verfahren aus (Jäkel, 2013) erwähnt, das in Abschnitt 3.2.2 beschrieben wurde. Für solche Planer muss es möglich sein, das auf der abstrakten Ebene erzeugte Wissen über die Zielaktion automatisch an die vom Manipulationsplaner erzeugte Aktion anzupassen.

Wissen über zu planende Aktionen Für Aktionen, die zum Erzeugen einer strukturierten Umgebung geplant werden, müssen deren Auswirkungen auf die Szene bekannt sein. Zum einen werden diese benötigt, um eine zielführend Aktion auswählen zu können, die eine Eigenschaft einer Szene erzeugt. Zum anderen ermöglicht es das Modell, den Zu-

stand der Umgebung nach der Ausführung einer Aktion vorherzusagen, was wiederum notwendig ist, um mehrere Aktionen, die verschiedene Effekte bewirken, zu einer Sequenz zu verketten. Eine solche Aktionssequenzen kann die geforderte Strukturiertheit der Zielszene erreichen.

Wissen über die Szene Um zu verändernde Eigenschaften einer Szene zu identifizieren wird ein Modell der Szene benötigt. Dieses muss kompatibel mit den verwendeten Aktionsmodellen sein. Das Modell der Szene muss aus der Wahrnehmung ebendieser erzeugt werden. Damit kann es erst zum Zeitpunkt der Ausführung bekannt sein. Daraus folgt die *Online-Fähigkeit* als wesentliche Anforderung an Modelle und Verfahren in dieser Arbeit.

Die zentrale Aufgabenstellung dieser Arbeit ist es, aus den im vorigen beschriebenen Wissen zur Laufzeit eine Sequenz von Aktionen zu erzeugen. In Kapitel 2 wurden drei Methoden vorgestellt, die in der Lage sind, Aktionssequenzen zu erzeugen: symbolische Planung, hierarchische Aufgabennetzwerke und POMDP. Hierarchische Aufgabennetzwerke bieten nur in Kombination mit einem symbolischen Planer die benötigte Flexibilität, Aktionssequnezen für unbekannte Szenen mit komplexen Zusammenhängen zu erzeugen. POMDP erzeugen eine Entscheidungsregel, die offline berechnet wird. Dies ist jedoch sehr aufwändig, aktuelle Ansätze verwenden deshalb eine grobe Diskretisierung, um den Rechenaufwand zu reduzieren. Diese ist wieder von Aspekten der Szene abhängig, welche im hier betrachteten Szenario erst zur Laufzeit bekannt ist. POMDP sind daher ebenfalls ungeeignet.

Stattdessen werden in dieser Arbeit symbolische Planer betrachtet. Diese werden aufgrund ihrer Fähigkeit, variable Aktionssequenzen zu erzeugen, gewählt. Die Länge der Pläne ist durch den Planer nicht beschränkt. Neue Aktionen können aufgrund von Anforderungen dem Plan hinzugefügt werden. Symbolische Planer können ausgehend von geeigneter Modellen der Szene Pläne für beliebige Szenenkonfigurationen erzeugen. Allerdings basiert symbolische Planung auf einer Suche in einem im Allgemeinen exponentiell wachsenden Zustandsraum. Die Laufzeit des Planers ist daher eine Herausforderung für die Online-Fähigkeit, die beim Entwurf des Systems berücksichtigt werden muss.

Durch die Wahl eines symbolischen Planers ergibt sich, dass die oben aufgeführten Modelle des benötigten Wissens in Form symbolischer Modelle dargestellt werden müssen, auf denen ein symbolischer Planer arbeiten kann. Innerhalb der Modelle findet die Prädiktion des Zustands statt. Jedoch ist die symbolische Domäne in ihren Möglichkeiten zur Beschreibung von Zuständen und Aktionen limitiert. Auch ist aufgrund der Echzeitanforderungen der Umfang der Planungsdomäne begrenzt. Das zu entwerfende symbolische Modell kann daher nur eine Annäherung an die Wirklichkeit sein. Die Konsequenzen daraus werden im nächsten Abschnitt formell analysiert.

4.3. Formales Metamodell: Wirklichkeit, Modell und Abstraktion

Um Vorhersagen über die Wirklichkeit treffen zu können, benötigt man Modelle von dieser. In dieser Arbeit werden auf oberster Abstraktionsebene symbolische Modelle verwendet. Grundlegende Eigenschaften der Beziehung zwischen Wirklichkeit und Modell werden im Folgenden aufgrund eines von Goos (2005) aufgezeigten Formalismus entwickelt. Dieser Formalismus definiert ein Modell von Modellen der Wirklichkeit. Um Begrifflichkeiten eindeutig zu halten, wird es im Folgenden mit dem Begriff *Metamodell* bezeichnet. Goos definiert dazu:

> **Definition 4.1** ***Wirklichkeit*** R *Dinge, Personen, Abläufe in der Zeit, Beziehungen zwischen diesen Gegenständen*

> **Definition 4.2** ***Modell*** M *Begriffe von (real existierenden oder nur gedachten) Dingen, Personen, Abläufen in gedachter Zeit, Beziehungen zwischen diesen Begriffen.*

Damit stellt er folgendes Diagramm auf:

$$\begin{array}{ccc} M & \xrightarrow{f_M} & M' \\ {\scriptstyle i}\uparrow & & \uparrow{\scriptstyle i} \\ R & \xrightarrow{f_R} & R' \end{array} \tag{4.1}$$

In dem Diagramm beschreibt i eine Abbildung, die Entitäten aus der Wirklichkeit R Begriffen des Modells M zuordnet. f beschreibt eine Beziehung zwischen den Entitäten über die Zeit. Damit das Modell als wahr betrachtet wird, muss das Diagramm 4.1 kommutieren, es muss also gelten:

$$f_M \circ i = i \circ f_R \tag{4.2}$$

Bei f_M handelt es sich um die Prädiktion des Modells:

Definition 4.3 ***Prädiktion*** *Die Prädiktion beschreibt die Überführung des Modells in das Modell, das zu einem zukünftigen Zeitpunkt gelten wird.*

4.3.1. Abstraktionsebenen

Diese Arbeit erweitert das Metamodell im Folgenden, um die Vorgänge bei Planungsaufgaben und deren Interaktion mit der Wirklichkeit darstellen zu können. Goos führt auf, dass die Abbildung i in mehreren Schritten durchgeführt werden kann. Damit entstehen mehrere Modellebenen, die im folgenden mit W bezeichnet werden.

Definition 4.4 ***Zustand*** W *Ein Zustand bezeichnet die Konfiguration der Welt, oder ein Modell davon zu einem Zeitpunkt.*

Die Abstraktionebene, in der ein Zustand definiert wird, wird als Index angehängt: W_n. W_0 entspricht dabei der Wirklichkeit, mit steigendem n werden die Modelle abstrakter, mit fallendem konkreter. Entsprechend werden die Abbildungen zwischen den Modellen mit *Abstraktion* A und *Konkretisierung* K bezeichnet.

Definition 4.5 ***Abstraktion*** A *Abbildung von einem konkreten zu einem abstrakteren Zustand:*

$$A : W_n \to W_m, m > n$$

Definition 4.6 ***Konkretisierung*** K *Abbildung von einem abstrakten zu einem konkreteren Zustand:*

$$K : W_m \to W_n, m > n$$

Im folgenden werden nur die Abbildungen zwischen benachbarten Abstraktionsebenen betrachten, für die $m - n = 1$ gilt. Es ergibt sich das Diagramm:

$$\begin{array}{ccc}
W_n & \xrightarrow{f_{W,n}} & W'_n \\
{\scriptstyle A_n}\uparrow\downarrow{\scriptstyle K_n} & & {\scriptstyle A_n}\uparrow\downarrow{\scriptstyle K_n} \\
W_{n-1} & \xrightarrow{f_{W,n-1}} & W'_{n-1} \\
{\scriptstyle A_{n-1}}\uparrow\downarrow{\scriptstyle K_{n-1}} & & {\scriptstyle A_{n-1}}\uparrow\downarrow{\scriptstyle K_{n-1}} \\
\vdots & & \vdots \\
{\scriptstyle A_1}\uparrow\downarrow{\scriptstyle K_1} & & {\scriptstyle A_1}\uparrow\downarrow{\scriptstyle K_1} \\
W_1 & \xrightarrow{f_{W,1}} & W'_1 \\
{\scriptstyle A_0}\uparrow\downarrow{\scriptstyle K_0} & & {\scriptstyle A_0}\uparrow\downarrow{\scriptstyle K_0} \\
W_0 & \xrightarrow{f_{W,0}} & W'_0
\end{array} \tag{4.3}$$

Die Abstraktion stellt dabei im Allgemeinen eine Vereinfachung eines Modells auf Eigenschaften, die in Bezug auf eine Problemstellung eine Bedeutung besitzen, dar. Die Abstraktion ist nicht injektiv, es existieren also unterschiedliche Modelle, die auf die selbe Abstraktion abgebildet werden. Dies ist auf den niedrigeren Informationsgehalt in abstrakteren Modellen zurückzuführen. Mit einem geeignet gewählten Modell kann die Abbildung surjektiv sein. Aufgrund

der fehlenden Injektivität ist sie nicht umkehrbar. Wird ein Modell W_n zuerst abstrahiert und dann wieder konkretisiert, so ist das Ergebnis i. A. nicht W_n:

Lemma 4.1 ***Unumkehrbarkeit der Abstraktion**:*

$$\exists W_n : (K_n \circ A_n)(W_n) \neq W_n$$

Der Umgekehrte Fall muss jedoch verlangt werden, d.h: Wird eine abstrakte Darstellung konkretisiert und wieder abstrahiert, so führt die Verkettung im Abstrakten wieder auf das selbe Modell:

Lemma 4.2 ***Umkehrbarkeit der Konkretisierung**:*

$$\forall W_n : (A_n \circ K_n)(W_n) = W_n$$

Weiter existiert für ein einzelnes W_n eine Menge U_{n-1}, für die gilt:

$$\forall W \in U_{n-1} : A_{n-1}(W) = W_n \tag{4.4}$$

Damit lässt sich für W_n eine Äquivalenzrelation definieren:

$$W_n \overset{A_n}{\sim} W'_n \Longleftrightarrow A_n(w_n) = A_n(W'_n) \tag{4.5}$$

mit der dazugehörigen Äquivalenzklasse

$$[W_n]_{A_n} := \{y \in W_n | y \overset{A_n}{\sim} W_n\} \tag{4.6}$$

Dies bedeutetet, dass auf der Abstraktionsebene n keine Möglichkeit besteht, Modelle der Ebene $n-1$ zu unterscheiden, wenn sie sich in der selben Äquivalenzklasse befinden. Unterschiede in konkreteren Ebenen können jedoch durchaus Konsequenzen für auf die Realität besitzen. Sie müssen auf den unteren Ebenen berücksichtigt werden.

In dieser Arbeit wird besonders der aufgrund der Verwendung eines symbolischen Planers notwendige Übergang zwischen kontinuierlicher und symbolischer Modellebene betrachtet. Es ergeben sich drei Abstraktionsebenen im Metamodell. Die Wirklichkeit W_w, das kontinuierliche Modell W_k und das symbolische Modell W_s. Davon unberührt bleibt die Tatsache, dass innerhalb dieser Modelle weitere Abstraktionsebenen existieren können. Ebenso können auch die Abbildungen zwischen den Ebenen mehrschichtig sein.

Definition 4.7 ***Symbolische Abstraktionsebene*** W_s *Modell aufgrund einer endlichen oder abzählbar unendlichen Menge an Werten.*

Sie ist die abstrakteste verwendete Darstellung. Es handelt sich um eine diskrete Darstellung. Um die endlichen Symbole zu nutzen, wird ihnen eine Semantik zugewiesen, sodass sie eine interpretierbare Bedeutung haben. Insbesondere bedeutet dies für Eigenschaften, die im Modell dargestellt werden, dass diese nur für einzelne Zeitpunkte definiert sein können und auf einen diskretisierten Wertebereich beschränkt sind. Im kontinuierlichen Modell W_k besteht diese Einschränkung nicht.

Definition 4.8 ***Kontinuierliches Modell*** W_k *Modell auf einer Abstraktionsebene, die kontinuierliche Wertebereiche zulässt.*

Die vorgeschlagene Aufteilung hat Ähnlichkeiten mit der klassischen Dreischichtenarchitektur (DSA) nach (Gat, 1997). Diese unterteilt Algorithmen nach deren Umgang mit Zuständen in drei Schichten: die unterste Schicht ist zustandslos, die mittlere Schicht besitzt Zustände, welche Vergangenheit und Gegenwart modellieren, die oberste Schicht kann auch in die Zukunft hinein prädizieren. Im hier verwendeten Metamodell verlaufen die Grenzen orthogonal zu denen in der DSA: In der oberen Schicht der DSA befindet sich das symbolisches Modell und die Methoden im kontinuierlichen Raum, die eine Prädiktion besitzen. Die unteren Schichten der DSA fallen in die kontinuierlichen Schicht.

Besondere Bedeutung für die Robotik haben die Abbildungen zwischen Wirklichkeit und kontinuierlichem Modell. Die Abstraktion ist in diesem Fall gleichbedeutend mit der Perzeption. So bildet die Perzeption einer Kamera ein Objekt an einer Pose in der Szene auf einen Objektnamen und eine numerische sechsdimensionale Darstellung der Pose ab.

In Richtung der Konkretisierung wird die Abbildung in die Wirklichkeit durch die Ausführung, bzw. die Aktuatorik, übernommen. So bildet ein Manipulator eine Trajektorie und Regelungsparameter auf eine konkrete Bewegung ab, sowie auch auf Kräfte und Momente an seinen Gelenken bzw. an Objekten mit denen er in Kontakt kommt. An diesem Beispiel ist noch einmal gut zu erkennen, das in der konkreten Ausführung deutlich mehr Zusammenhänge wirken, als in der abstrakten Darstellung modelliert wird. So entstehen z.B. auch für eine im Abstrakten rein kinematisch definierte Trajektorie in den Gelenken des Armes Momente.

Die Abbildung zwischen den unteren Ebenen gehört zu den Grundaufgaben der Robotik und die Nomenklatur ist dementsprechend verbreitet. Für die Verbindung symbolischer mit kontinuierlicher Repräsentation wird der Begriff „Symbolverankerung"[1] verwendet. Er ist eng mit dem „Symbolverankerungsproblem" (Harnad, 1990) verknüpft, welches Symbolen eine konkrete Bedeutung zuordnet. Das Symbolverankerungsproblem wird in dieser Arbeit vermieden, da Symbole direkt aus der Szene erzeugt werden und nicht aus zusätlichen Wissen in das Modell

[1] engl.: symbol grounding

kommen. Hier interessiert der umgekehrte Fall; es wird daher der Begriff der *Symbolisierung* für die Abstraktion vom kontinuierlichen ins abstrakte Modell eingeführt.

Definition 4.9 ***Symbolisierung*** S *Die Symbolisierung ist die Abbildung von einem kontinuierlichen in ein symbolisches Modell.*

Für die umgekehrte Richtung, also die Konkretisierung vom Symbolischen zum Kontinuierlichen kennt die Literatur zwar eine Vielzahl von Verfahren (s. Anhang A), es ist kein gängiger Name verbreitet. Es wir in den folgenden Kapiteln der Begriff Konkretisierung ausschließlich für diesen Fall der Konkretisierung verwendet. Für die Konkretisierung von kontinuierlichen Modell zur Wirklichkeit wird der Begriff der Ausführung verwendet.

4.3.2. Zeitlicher Verlauf

Der Zustand der Umgebung soll zielgerichtet verändert werden. Dazu ist es nötig, in der Welt zielgerichtet eingreifen zu können. Dies geschieht durch Aktionen:

Definition 4.10 ***Aktion*** *Eine Aktion ist eine atomare, steuerbare, initierbare Zustandsänderung eines Agenten in der Wirklichkeit. Sie hat eine zeitliche Ausdehnung und ist auf ein Ergebnis ausgerichtet.*

Die Aktion ist auf der Abstraktionsebene der Wirklichkeit definiert. Auf abstrakteren Ebenen werden Aktionen durch *Operatoren* OP dargestellt. Ein Operator prädiziert die Abstraktion der durch Anwendung der Aktion entstehenden Wirklichkeit.

Um zielgerichtet zu planen, benötigt man eine Vorhersage der Welt in die Zukunft. Die Modelle werden dafür um eine zeitliche Komponente erweitert. Das Modell W zu einem Zeitpunkt t wird zum *Weltzustand* W^t. Die Beziehung $f_{W,i}$ kann durch Operatoren ersetzt werden. Damit ergibt sich das folgende Diagramm:

$$
\begin{array}{ccccccc}
W_s^0 & \xrightarrow{\mathrm{OP}_s^0} & W_s^1 & \xrightarrow{\mathrm{OP}_s^1} & \dots & \xrightarrow{\mathrm{OP}_s^{n-1}} & W_k^m \\
\updownarrow{\scriptstyle S}\,\updownarrow{\scriptstyle K} & & \updownarrow{\scriptstyle S}\,\updownarrow{\scriptstyle K} & & \updownarrow{\scriptstyle S}\,\updownarrow{\scriptstyle K} & & \updownarrow{\scriptstyle S}\,\updownarrow{\scriptstyle K} \\
W_k^0 & \xrightarrow{\mathrm{OP}_k^0} & W_k^1 & \xrightarrow{\mathrm{OP}_k^1} & \dots & \xrightarrow{\mathrm{OP}_k^{n-1}} & W_k^m \\
\updownarrow{\scriptstyle P}\,\updownarrow{\scriptstyle X} & & \updownarrow{\scriptstyle P}\,\updownarrow{\scriptstyle X} & & \updownarrow{\scriptstyle P}\,\updownarrow{\scriptstyle X} & & \updownarrow{\scriptstyle P}\,\updownarrow{\scriptstyle X} \\
W_w^0 & \xrightarrow{\mathrm{OP}_w^0} & W_w^1 & \xrightarrow{\mathrm{OP}_w^1} & \dots & \xrightarrow{\mathrm{OP}_w^{n-1}} & W_w^m
\end{array}
\tag{4.7}
$$

Dabei bezeichnet X die Ausführung als Spezialfall der Konkretisierung, P die Perzeption als Spezialfall der Abstraktion. Im symbolischen Modell gibt es nur abzählbar unendlich viele Zu-

stände. Sie können mittels Operatoren ineinander überführt werden. Es lässt sich damit ein Zustandsgraph aller symbolischen Weltzustände W_s aufbauen. In diesem kann mittels einer Suche eine Sequenz von Operatoren gefunden werden, die einen symbolischen Startzustand in einen Zielzustand überführt. Eine solche Sequenz bezeichnet man als *Plan*, die Suche nach diesem als *Planung* bzw. als klassisches oder symbolisches Planungsproblem.

Definition 4.11 ***Plan*** *Eine Sequenz von Aktionen, deren zugeordnete Operatoren einen Startzustand in einen Zielzustand überführen.*

$$\begin{aligned} \text{Planung} : (W_s^0, W_s^n) \mapsto (OP_s^0, OP_s^1, \ldots, OP_s^{n-i}) \text{ mit} \\ OP_s^{n-1} \circ \ldots \circ OP_s^1 \circ OP_s^0(W_s^0) = W_s^n \end{aligned} \tag{4.8}$$

Das klassische Planungsproblem ist ein gut erforschtes Problem der künstlichen Intelligenz (KI). Der Stand der Technik dazu wird in Abschnitt 2.2 dieser Arbeit dargelegt. Er wird auch in einführenden Lehrbüchern der KI wie (Russell u. Norvig, 2003) ausgiebig behandelt. In dieser Arbeit interessiert ein erweitertes Problem. Zur Handlungsadaption für einen Roboter genügt es nicht, Pläne nur in der symbolischen Abstraktionsebene auszuführen. Stattdessen ist ein der Anfangszustand W_w^0 durch die Wirklichkeit gegeben, dieser wird durch Perzeption und Sensorik auf einen symbolischen Zustand W_s^0 abgebildet. Auch der Zielzustand W_s^n ist durch Eigenschaften eines Zustandes im Kontinuierlichen oder Symbolischen gegeben. Das Ziel der Handlung muss es letztlich sein, einen zu W_s^n konsistenten Zustand der Wirklichkeit W_w^n zu erreichen. Die Wirklichkeit kann im beschriebenen Metamodell ausschließlich durch das Ausführen von Aktionen beeinflusst werden, d. h. alle generierten symbolischen Operatoren müssen vom Symbolischen in das Kontinuierliche abgebildet und in der Wirklichkeit ausgeführt werden. Zur Lösung eines Handlungsplanungsproblem muss, analog zu Diagramm (4.7) mindestens folgendes Diagramm aufgebaut werden:

$$\begin{array}{ccccccc} W_s^0 & \xrightarrow{OP_s^0} & W_s^1 & \xrightarrow{OP_s^1} & \ldots & \xrightarrow{OP_s^{n-1}} & W_k^n \\ \uparrow{\scriptstyle S} & & \downarrow{\scriptstyle K(OP_s^1)} & & \downarrow{\scriptstyle K(OP_s^{n-1})} & & \\ W_k^0 & & W_k^1 & & \ldots & & \\ \uparrow{\scriptstyle P} & & \downarrow{\scriptstyle X(OP_k^1)} & & \downarrow{\scriptstyle X(OP_k^{n-1})} & & \\ W_w^0 & \xrightarrow{A^0} & W_w^1 & \xrightarrow{A^1} & \ldots & \xrightarrow{A^{n-1}} & W_w^n \end{array} \tag{4.9}$$

Dabei bezeichnet A_0 eine leere Aktion, also die physische Untätigkeit des Roboters während der Planung. Zu erwarten, aber nicht zwingend, sind auch Abbildungen zwischen den kontinu-

ierlichen Modellen, da diese wie festgestellt nicht vollständig aus dem symbolischen Modell erstellt werden können. Es werden neben einem symbolischen Planer insbesondere benötigt:

- Geeignete Zustandsbeschreibung W_s im Symbolischen und W_k im Kontinuierlichen. Insbesondere im Symbolischen erfolgt die zur Planung notwendige Prädiktion ausschließlich aufgrund des im symbolischen Modell kodierten Wissens.
- Eine geeignete Operatorenmenge sowie eine Beschreibung der Anwendbarkeit der Operatoren und der durch die Operatoren erzeugten Zustandsübergänge (vgl. Kap. 5)
- Geeignete Abbildungen zur Abstraktion ins Symbolische: Symbolisierung S (vlg. Kap. 6)
- Konkretisierungen und Ausführungen für Operatoren (vgl. Kap. 7)

Sind diese vorhanden, so kann das Handlungsplanungsproblem

$$\begin{aligned} \text{Handlungsplanung} : (W_w^0, W_s^n) \mapsto (A^0, A^1, \ldots, A^{n-i}) \text{ mit} \\ S(P(A^{n-1} \circ \ldots \circ A^1 \circ A^0(W_w^0))) = W_s^n \\ A^i = X(K(OP_s^i)) \end{aligned} \tag{4.10}$$

auf das symbolische Planungsproblem aus Gleichung (4.8) abgebildet werden.

4.3.3. Grenzen des Metamodells

Dem im vorigen Abschnitt dargestellten Metamodell für Handlungsplanung sind in der praktischen Anwendung Grenzen gesetzt. Seine Annahmen können nicht immer vollständig erfüllt werden. Der Prozess der Abstraktion erlaubt es, die Welt in eine Darstellung zu bringen, in der Schlüsse und Prädiktionen möglich sind, jedoch auf Kosten der Vollständigkeit des Modells. Folgenden Punkte beeinflussen die Korrektheit der Prädiktion negativ:

Uneindeutige Perzeption Sensoren bilden die Wirklichkeit auf ein Modell ab. Durch physikalische Effekte besitzen sie jedoch nur eine begrenzte Genauigkeit, die Wirklichkeit eines Messwertes befindet sich innerhalb gewisser Schranken um den Wert. Dies kann modelliert werden, indem der Messwert m als $m = w + r$ dargestellt wird. Dabei bezeichnet w den wirklichen Wert und r einen Rauschanteil, der zufällig ist. Damit kann für zwei Messungen der Fall auftreten, dass die Perzeption zwei gleichen Weltzustände $W_w^n = W_w^m$ unterschiedliche kontinuierliche Weltzustände $W_k^n \neq W_k^m$ zuordnet.

Unvollständige Abstraktion Die Abstraktion führt eine Reduktion der Zustände ein. Dadurch existieren immer Aspekte, die in einer niedrigen Ebene dargestellt werden, in der abstrakten jedoch fehlen. Sobald solch ein Aspekt eine relevante Eigenschaft beeinflussen kann, ergibt sich eine Inkonsistenz im Modell. Als Beispiel hierfür können Objektgeome-

trien dienen. Für Bahnplanungsaufgaben werden diese angenähert, sodass eine einfache Geometrie die ursprüngliche vollständig einschließt. Damit kann ein Bahnplaner Trajektorien generieren, die sicher kollisionsfrei sind, jedoch ist es auch möglich, das die Annäherung Trajektorien verhindert, die in der Wirklichkeit ausführbar sind.

Zeitliche Diskretisierung Ein besonders hervorzuhebender Aspekt der unvollständigen Abstraktion ist die zeitliche Diskretisierung. Sie hat zur Folge, dass es Zeitpunkte im Modell gibt, die in der Wirklichkeit und im kontinuierlichen Modell existieren, im symbolischen aber nicht. Ereignisse zwischen denen von den abstrakten Operatoren vorgegebenen Zeitpunkten können nicht modelliert werden.

Unvollständige Operatoren Im Symbolischen können die Operatoren nicht jeden Effekt ihrer Ausführung in der Wirklichkeit abbilden. z.B. kann das Greifen eines Kartons dazu führen, dass dieser nicht nur seinen Ort verändert, sondern durch die Krafteinwirkung auch deformiert wird.

Nicht lineare Fehlerfortschreibung Kleine Fehler in den Modellen, etwa aus der verrauschten Sensorik, die im Modell eines einzelnen Zustandes in akzeptablen Grenzen liegen, können durch die Prädiktion von Folgezuständen zu nicht linear wachsenden, großen Fehlern anwachsen. Wird etwa ein Objekt minimal neben seiner Position lokalisiert, kann es bei anschließenden Greifen im Greifer verrutschen, dadurch könnte es z.B. verdreht abgestellt werden. Wird aus dem prädizierten Modell nun eine neue Greif-Operation generiert, so kann der Griff des Manipulators ins Leere gehen. Das Modell besitzt nun einen sehr großen Fehler bezüglich der Position des Objektes.

Offenes System Änderungen des Weltzustandes können im Modell nur aus dem Modell heraus oder durch das Anwenden eines Operators geschehen. Es wird ein geschlossenes System angenommen. In der Wirklichkeit können jedoch Ereignisse eintreten, die nicht dem betrachteten System entstammen. So kann ein leichtes Objekt seine Position auch durch eine Erschütterung verändern, oder ein Mensch greift unerwartet ein. Da es nicht möglich ist, jeden Aspekt der Wirklichkeit zu modellieren und vorherzusagen, muss ein offenes System angenommen werden, in dem Störungen von außen auftreten können.

Diese Einschränkungen führen dazu, dass in jeder real umsetzbaren Implementierung das Diagramm (4.9) nicht kommutiert. Für gegebene Operator-Sequenzen wird die Fortschreibung des Weltzustandes auf den Abstraktionsebenen zu zueinander inkonsistenten Ergebniszuständen führen. Ein Ziel dieser Arbeit ist es daher, ein Konzept zu entwickeln, welche die Konsistenz eines Plans und seiner Ausführung sicherstelt.

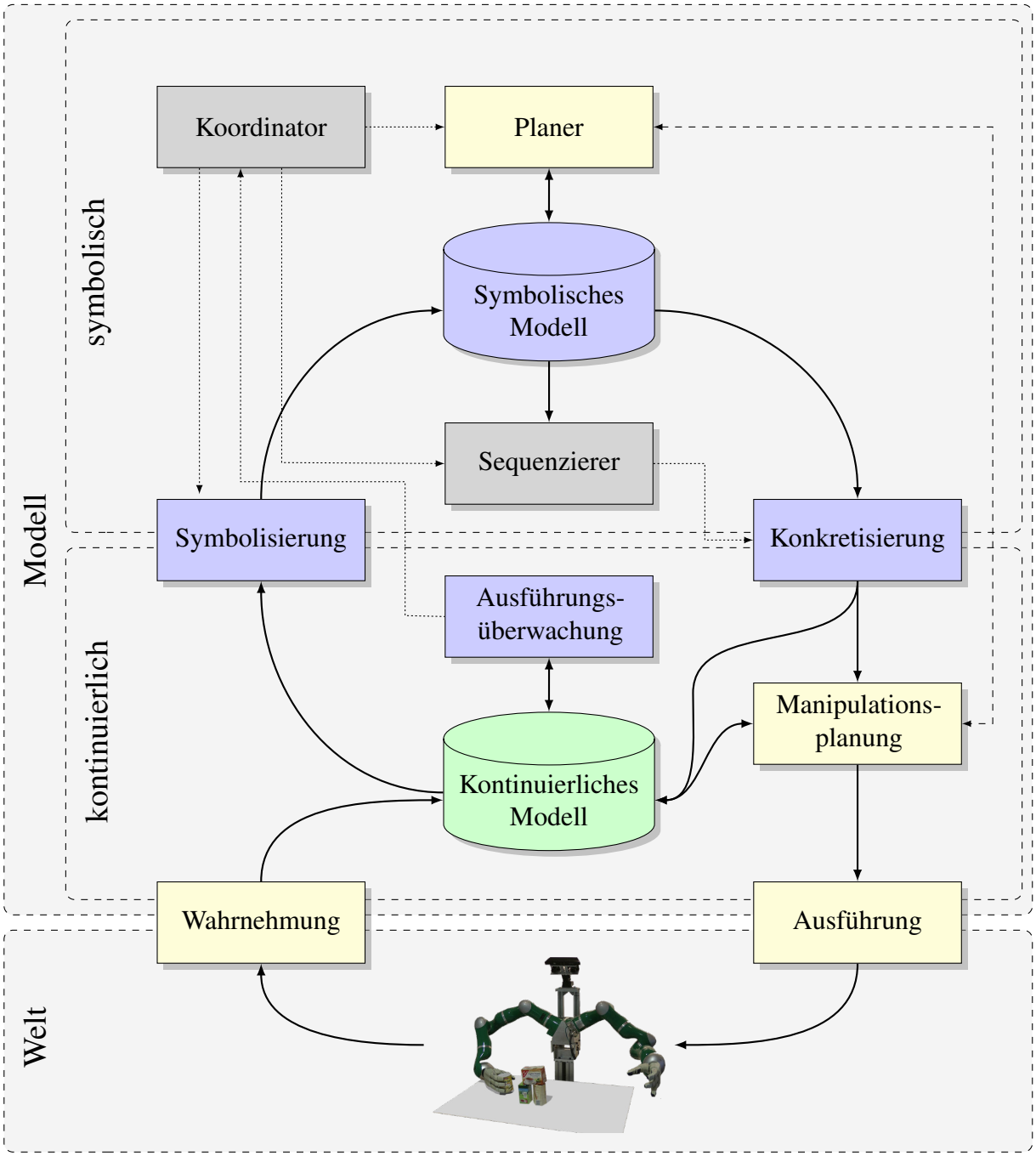


Abb. 4.1.: Überblick über die Architektur des Systems. Auf der linken Seite sind die Abbildungen der Abstraktion zu finden. Sie abstrahieren die Szene, dargestellt durch einen manipulierenden Roboter, über des kontinuierliche Modell zu ihrer symbolische Beschreibung. Auf der rechten Seite sind die Abbildungen der Konkretisierung dargestellt, die letztlich zu einer Ausführung und damit einer neuen Szene führen. Durchgehende Pfeile stellen den Datenfluss dar, gepunktete und gestrichelte eine Auslösung bzw. Anfrage. Komponenten, zu denen ein Beitrag in dieser Arbeit geleistet wird, sind blau hinterlegt. Gelbe Komponenten entstammen dem Stand der Technik. Graue Komponenten organisieren das Zusammenspiel und sind wissenschaftlich von untergeordneter Bedeutung.

4.4. Ansatz zur Handlungsadaption

Aufgrund der Ergebnisse der vorausgegangenen Problemanalyse wird in diesen Kapitel ein System zur Adaption von Handlungssequenzen entwickelt. Aus den bei der Problemanalyse angestellten Überlegungen ergibt sich, dass folgende Komponenten benötigt werden, um im beschriebenen Metamodell den vollständigen Bogen von der Szene hin zur geplanten Ausführung in der Szene zu spannen:

- Wahrnehmung
- Symbolisierung von Szenen
- Symbolische Modellierung von Aktionen
- Symbolische Handlungsplanung
- Konkretisierung geplanter symbolischer Aktionen
- Ausführung

Das Gesamtbild des Systems fügt sich zu einer Sense-Plan-Act Architektur (Nilsson, 1984). Aus den Überlegungen in Abschnitt 4.3.3 folgt allerdings, dass diese für die gewählte Lösung durch symbolische Planung nicht ausreichend ist. Um das Ziel der Konsistenz zwischen Plan und Ausführung zu überwachen und basierend auf der Überwachung, umzusetzen, wird eine weitere Komponente eingeführt:

- Ausführungsüberwachung

Die Komponenten werden zu einem Gesamtsystem zusammengesetzt. Dieses ist in Abb. 4.1 dargestellt. Die Architektur stellt ein hierarchisch auf zwei Planungsschichten ausgedehntes Sense-Plan-Act System mit Erweiterung zur Neuplanung in Fehlerfall dar. Die Verbindung zwischen symbolischer Planung und Manipulationsplanung geht über klassische Ansätze solcher hierarchisch aufgeteilten Planungssysteme hinaus. Böcke stellen in dem Diagramm benötigte Abbildungen dar. Sie werden durch ein oder mehrere Verfahren implementiert. Durchgehende Pfeile beschreiben den Datenfluss. Gepunktete Linen Visualisieren den Kontrollfluss des Koordinators. Die folgende Erklärung beschreibt das Zusammenspiel der Komponenten. Sie orientiert sich dabei am Datenfluss von der Szene zur Planung und zurück zur Ausführung.

Ausgehend von der Szene bildet die *Wahrnehmung* die Wirklichkeit in das kontinuierliche Modell ab. Sie verwendet dazu optische Sensoren, hauptsächlich aus dem Bereich der 3D Sensorik. Algorithmen bereiten die wahrgenommenen Informationen weiter auf. Am Ende der Verarbeitung werden sie im kontinuierlichen Modell gespeichert. Die Wahrnehmung wird in kurzen Zyklen ausgeführt, um das kontinuierliche Modell aktuell zu halten. Die unverarbeiteten Sensorwerte gehen in Echtzeit in das kontinuierliche Modelle ein; dadurch wird mit der Ausführungsüberwachung eine Fehlererkennung in Echtzeit ermöglicht.

Das *kontinuierliche Modell* beschreibt das dem System auf dieser Abstraktionsebene zur Verfügung stehende Wissen. Es beinhaltet neben dem online erzeugen Wissen aus der Szene auch Hintergrundwissen über Objekte, Wissen über den Roboter und Aktionen. Es deckt damit alle Wissensbereiche aus der Problemanalyse ab; insbesondere folgende Aspekte:

Szene Objektpositionen, Objektgeometrienen, Visuelle Modelle, Hindernismodelle, Objektphysik (Masse, Schwerpunkt, Reibung)

Roboter Kinematisches Modell, geometrisches Modell, visuelles Modell, Modell der Roboterfähigkeiten

Aktionen Griffe, Trajektorien, Einschränkungen zur Erfolgsbewertung, Roboterkonfigurationen.

Das im kontinuierlichen Modell vorhandene Wissen wird durch den Prozess der *Symbolisierung* in das symbolische Modell überführt. Die Kernaufgabe diese Prozesses ist es, das kontinuierliche Wissen auf ein möglichst vollständiges Modell aus endlich vielen Symbolen mit beherrschbarer Komplexität abzubilden. Es werden die unterschiedlichen, die in Abschnitt 4.2 identifizierten Aspekte, behandelt. Sie führen zu verschiedenartigen Symbolen. Daher wird auf eine Menge von Verfahren zurückgegriffen. Diese Implementieren die Symbolisierungsoperatoren. Dabei ist ein Symbolisierungsoperator für jeweils eine Art von Symbolen zuständig (beispielsweise „An Ort" zur Beschreibung des Ortes). Da die Symbolisierung ein rechenaufwändiger Prozess ist, wird sie nur bei Bedarf vor der symbolischen Planung durchgeführt. Sie wird ausführlich in Kap. 6 beschrieben.

Die Ergebnisse der Symbolisierung werden als Vereinigungsmenge zum *symbolischem Modell* zusammengefügt. Auch dieses muss die in Abschnitt 4.2 identifizierten Wissensaspekte enthalten. Seine Modellierung ist eng mit dem verwendeten *Planer* verknüpft. In dieser Arbeit wird ein auf Zeitstrahlen basierender Planer verwendet, der die Verteilung von Aktionen auf ausführende Ressourcen ermöglicht (s. Kap. 5). Wie im kontinuierlichen Modell lässt sich das Wissen im symbolischen Modell in die Bereiche Szene, Roboter und Aktionen untergliedern. Zusätzlich werden Informationen über zukünftige Aktionen, also Pläne und resultierende Szenen gespeichert. Dieses Wissen entsteht aus dem symbolischen Planer.

Neben der engen Anbindung an das symbolische Modell besitzt der Planer auch noch eine lose Verbindung zu den *Manipulationsplaneren*. Um Effekte des unvollständigen symbolischen Modells auszugleichen, werden diese verwendet, um die prizipielle Ausführbarkeit von Aktionen in einer Szene sicherzustellen. Erfolgreiche Anfragen an den Manipulationsplaner können dadurch im symbolischen Modell als Vorbedingung verwendet werden.

Der *Koordinator* koordiniert die Planung. Dazu stößt er die Symbolisierung an, im Anschluss den Planer und bei dessen Erfolg löst er die Abarbeitung des Plans durch den Sequenzierer (s.

nächster Abschnitt) aus. Während der Ausführung wird er von der Ausführungsüberwachung über Fehler informiert. Im Fehlerfall stoppt er die Ausführung durch den Sequenzierer und leitet die Neuplanung ein.

Der *Sequenzierer* wählt die aktuell auszuführende Aktion aus dem Plan im symbolischen Modell aus und veranlasst die Konkretisierung zur Abbildung des Aktionssymbols und seiner Parameter ins kontinuierliche Modell.

Die *Konkretisierung* benutzt Methoden der Manipulationsplanung wie Greifplanung und kollisionsfreie Bahnplanung, um für Aktionsymbole und deren Parameter Konkretisierungsoperatoren bereit zu stellen. Diese erzeugen eine Abbildung auf eine kontinuierliche Ausführung der Aktion. Meist ist dies eine Trajektorie. Die Konretisierungsoperatoren arbeiten dazu auf dem kontinuierlichen Modell, wodurch sie Zugriff auf das von der Wahrnehmung aktualisierte Weltmodell haben und nicht ausschließlich von der symbolischen Prädiktion der Szene abhängen. Hier kann es jedoch zu Inkonsistenzen kommen, sodass der Planer keine Lösung finden kann. In so einem Fall muss eine symbolische Neuplanung auf dem aktualisierten Modell durchgeführt werden. Um Inkonsistenzen frühzeitig zu erkennen, bildet die Konkretisierung auch die Ausführungsbedingungen ab, die von der Ausführungsüberwachung verwendet werden. Die Methoden der Konkretisierung werden in Kap. 7 behandelt.

Die *Ausführung* nutzt Verfahren der Interpolation und der Regelung, um Trajektorien auf einem Manipulator oder anderer Hardware in der Wirklichkeit auszuführen. Sie ist nicht Teil dieser Arbeit, aber doch essentieller Bestandteil des Systems, da sie letztlich den Verarbeitungszyklus schließt. Durch sie wird die Szene verändert.

Die *Ausführungsüberwachung* ist während der Ausführung einer Aktion für das Erkennen von Fehlern in Echtzeit zuständig. Dafür werden im kontinuierlichen Modell Ausführungsbedingungen gespeichert, deren Erfülltheitsgrad die Ausführungsüberwachung kontinuierlich überprüft. Bei Widersprüchen der Wahrnehmung zu den Ausführungsbedingungen wird der Koordinator informiert, der die Korrektur des Plans durch Neuplanung auslöst. Zu Details sei auf Kap. 7 verwiesen.

Der Block der *Manipulationsplanung* verwendet Methoden, die auch bei der Konkretisierung zum Einsatz kommen. Er arbeitet auf einem kontinuierlichen Modell, das nicht die Wirklichkeit darstellt, sondern aus der Prädiktion der symbolischen Planung entsteht. Damit kann die Manipulationsplanung die Ausführbarkeit zu einem zukünftigen Zeitpunkt im Plan bewerten. Sie kann dabei auf zusätzliches Wissen aus dem kontinuierlichen Modell zugreifen (etwa Objektmodelle) und damit eine bessere Entscheidung über die Anwendbarkeit einer Aktion treffen, als es der symbolische Planer rein auf seinem Modell basierend könnte.

4.5. Fazit

Dieses Kapitel hat sich ausgiebig mit der Analyse der Problemstellung dieser Arbeit beschäftigt. Es wird der Schluss gezogen, dass ein symbolischer Planer ein geeigneter Ansatz zur Adaption von Handlungssequenzen ist. Daher wurde er als Grundlage des Systems gewählt. Diese Entscheidung führt zu weitreichenden Konsequenzen und neuen Teilproblemen. In einer Analyse aufgrund einer formellen Beschreibung wurden relevante Komponenten und Anforderungen an ihre Umsetzung identifiziert.

Aufgrund der Analysen wurde ein System aus mehreren Teilkomponenten aufgezeigt und deren Zusammenspiel beschrieben. In den folgenden Kapiteln werden die Komponenten detailliert ausgearbeitet.

5. Symbolische Modelle und Handlungsplanung

Das in Kap. 4 vorgestellte Konzept basiert auf einem symbolischen Planer, der Entscheidungen für Aktionen aus dem symbolischen Modell heraus fällt. Das symbolische Modell ist ausschlaggebend für den Erfolg der Planung. Nicht modellierte Aspekte können nicht oder nur unter hohem Aufwand, durch Anfragen an Verfahren im Kontinuierlichen, berücksichtigt werden. Auf der anderen Seite beeinflusst ein zu umfangreiches symbolisches Modell die Laufzeit des Planers ungünstig. Im Folgenden werden daher Manipulationsaktionen sowie deren Eigenschaften und Anforderungen analysiert. Es werden relevante Eigenschaften des Weltzustands identifiziert. Aus ihnen wird ein geeignetes symbolisches Zustandsmodell, welches insbesondere das Szenenmodell enthält, entworfen.

Weiter beinhaltet das symbolische Modell die Aktionen des Serviceroboters und ihre Beschreibung. Sie werden in Abschnitt 5.3 vorgestellt. In Abschnitt 5.4 wird die Darstellung der entwickelten Modelle durch Zeitstrahlen entworfen. Eine angepasste Version des Plaungsframeworks EUROPA-PSO erlaubt die Implementierung des vorgestellten Konzeptes.

5.1. Identifizierung relevanter Aspekte von Manipualtionshandlungen

Das symbolische Modell stellt die für die Planung relevanten Informationen dem symbolischen Planer bereit. Sie beinhalten nach Abschnitt 4.2 die Zielaktionen, zu planende Aktionen und die Modellierung der Szene. Hier sollen vom Begriff der Aktion ausgehend relevante Aspekte für Manipulationshandlungen entwickelt werden. Eine allgemeine Beschreibung von Aktionen liefert die deutsche Sprache. Ein einfacher Satz wie „Roboter greift Dose" hat die Form: *Subjekt Prädikat Objekt*. Die beteiligten Entitäten müssen modelliert werden, um Aktionen beschreiben zu können. Im symbolischen Modell gilt:

1. *Ressourcen* modellieren das Subjekt als ausführendes Element. Dies kann der Roboter oder eine seiner Komponenten sein.
2. *Aktionen* modellieren das Prädikat. Sie sind zeitlich begrenzte Teilhandlungen mit definierten Effekten.
3. *Objekte* sind die Entitäten, auf die Aktionen angewandt werden.

Aktionen können im symbolischen Modell durch Vorbedingungen und Effekte beschrieben werden. Beide setzten die beschriebene Aktion in eine Relation zur Szene, in der diese ausgeführt wird. Vorbedingungen beschreiben Relationen, die vor der Ausführung einer Aktion erfüllt sein müssen, damit diese zur Anwendung kommen kann. Der Effekt beschreibt die Konsequenzen der Ausführung einer Aktion. Im Fall der Manipulationsplanung beinhaltet der Effekt eine Veränderung der Szene. Zu diesen für die Planung relevanten Aspekten einer Aktion hinzu kommen die Währendbedingungen, die während Ausführung erfüllt sein müssen. Damit das symbolische Modell die Relation einer Aktion zur Szene darstellen kann, muss die Modellierung der Szene im Symbolischen eng verknüpft mit der Modellierung von Aktionen betrachtet werden.

Folgende Vorbedingungen sind für Manipulationsaktionen relevant:

4. *Anwendbarkeit der Aktion auf das Objekt:* Beschreibt, ob eine Aktion auf ein Objekt anwendbar ist. Z. Bsp. muss zum Greifen eines Objekte ein passender Griff bekannt sein.
5. *Erreichbarkeit der Objekte:* Können Konfigurationen erzeugt werden, in denen das Objekt für den Roboter unter den Anforderungen der Aktion erreichbar ist? Aufgrund des Bezugs zur Kinematik spielen in diesem Punkt Aspekte aus der kontinuierlichen Ebene eine Rolle. Im Symbolischen wird die Information über den Ort benötigt, um die Fragestellung an die kontinuierliche Ebene zurückgeben zu können.
6. *Erreichbarkeit assoziierter Orte:* Wie im Vorigen, jedoch werden Orte in diesem Aspekt nicht durch Objekte spezifiziert, sondern durch die Aktion.
7. *Zustand assoziierter Objekte:* Der Zustand eines Objektes hat Einfluss auf die Anwendbarkeit einer Aktion. So muss etwa ein Flasche geöffnet sein, um aus ihr einschenken zu können.
8. *Zustand assoziierter Orte:* Ein Ablageort darf nicht durch ein Hindernis belegt sein.
9. *Relation von assoziierten zu anderen Objekten:* Durch räumliche oder mechanische Beziehungen zwischen Objekten kann die Durchführbarkeit von Aktionen beeinflusst werden, etwa wenn ein Hindernis in der Nähe des Objektes eine Greifaktion blockiert.
10. *Zustand des Roboters:* Es wird Wissen über den Zustand des Roboters benötigt. Damit eine Aktion durchgeführt werden kann, müssen benötigt Ressourcen des Roboters vorhanden und frei sein.

Während der Ausführung spielen folgende Währendbedingungen eine Rolle:

11. *Kollisionsfreiheit:* Berührungen des Roboters mit der Umwelt sind bei Manipulationsaktionen nur für die Objekte der Aktion erlaubt. Berührungen mit anderen Objekten führen zu unvorhersehbaren und potentiell unerwünschten Effekten auf die Umwelt. Durch entstehende Kräfte kann der Roboter an der präzisen Ausführung der Aktion gehindert

werden, soweit, bis der Roboter selbst Schaden nimmt. Die Vermeidung ungewollter Kollisionen ist daher ein wesentlicher Aspekt der Ausführung jeder Manipulationsaktion.

12. *Zustand des Roboters:* Während der Ausführung ändert sich der Zustand des Roboters. Dies führt zu den unter Vorbedingung 10 beschriebenen Konsequenzen.
13. *Zustand assoziierter Objekte:* So wie eine Komponente des Roboters durch die Verwendung für eine Aktion ihren Zustand ändert, gilt dies auch für manipulierte Objekte. Ein gegriffenes Objekt kann z.B. nicht ohne Weiteres durch den zweiten Greifer des Roboters gegriffen werden.

Zuletzt müssen die Effekte von Aktion beschreibbar sein:

14. *Ort assoziierter Objekte:* Aktionen können den Ort von Objekten verändern.
15. *Zustand assoziierter Objekte:* Aktionen können den Zustand von Objekten verändern.
16. *Zustand des Roboters:* Aktionen können den Zustand des Roboters verändern.
17. *Zustand weiterer Objekte* Die Ausführung einer Aktion kann weitere Objekte beeinflussen, selbst wenn diese für die Aktion selbst keine Rolle spielen. Wird ein tragendes Objekt bewegt, so beeinflusst das die Lage der getragenen Objekte. Dies kann ungewollt sein, wenn Objekte unkontrolliert fallen. Ebenso kann es gewollt sein, wie beim Transport eines Objektes auf einem Tablett.

Im folgenden Abschnitt wird ein symbolisches Modell entworfen, welches die beschriebenen Aspekte darstellen kann.

5.2. Zustandsmodell zur Planung von Manipulationsaktionen

In diesem Abschnitt werden die Symbole des symbolischen Modells untersucht, die zur Beschreibung des Weltzustandes benötigt werden. Das symbolische Modell wird als ganzes betrachtet. Es wird seine Vollständigkeit in Bezug auf die in Abschnitt 5.1 identifizierten Aspekte dargelegt.

Modelle zur symbolischen Planung erlauben zwei unterschiedliche Arten von Symbolen: zum einen Symbole, die Entitäten beschreiben und keine weiteren Abhängigkeiten haben. Zum anderen Relationen, die Beziehungen zwischen den Entitäten beschreiben, bzw. den Zustand einzelner Eigenschaften einer Entität. Entitäten werden nur durch ihren Namen beschrieben. Relationen besitzen einen Namen, zusätzlich werden die Entitäten, deren Beziehung sie beschreiben, in Klammern angehängt. Das Szenenmodell ist die Vereinigungsmenge der Symbole der Entitäten und der Relationen. Zusätzlich zu den Symbolen werden in dieser Arbeit semantische Anhänge (Dornhege u. a., 2010) verwendet. Sie erlauben es, Symbole mit Informationen aus

Symbolklasse	Beispiel	Bedeutung	Aspekt
Ressource	„Linker Manipulator”	Entität, die eine Aktion ausführen kann. Sie ist eine Komponente des Roboters	1
Aktionen	„Transport”	Eine Aktion wird aktiv vom Roboter ausgeführt und verändert den Zustand der Welt	2
Objekt	„Sauerkrautdose1”, „Obj1”	Ein Objekt in der Szene, auf das Aktionen angewandt werden können	3
Objekttyp	„Sauerkrautdose”,	Beschreibt ein Klasse von Objekten	
Ort	„Ort von Sauerkrautdose1”, „Ort 1”	Ein Ort in der Szene	6
Roboterkomponente	„Linker Arm”	Entität, die eine atomare Aktion ausführen kann	12

Tab. 5.1.: Überblick über die verwendeten Symbolklassen zur Darstellung von Entitäten. Die Entitäten werden durch die Relationen aus Tabelle 5.2 zum symbolischen Modell verknüpft.

dem kontinuierlichen Modell zu annotieren. Diese Informationen werden bei der Konkretisierung verwendet. Der symbolische Planer dagegen kann nicht auf sie zugreifen.

Tabelle 5.1 listet die Entitäten des symbolischen Modells auf. „Ressourcen”, „Aktionen” und „Objekte” ergeben sich direkt aus den Anforderungen zur allgemeinen Beschreibung einer Aktion als notwendige Entitäten. Die zusätzlich eingeführte Entität „Ort” wird aufgrund der manipulationsspezifischen Anforderungen benötigt. Viele der identifizierten Aspekte der Manipulation setzen unterschiedliche Eigenschaften und Entitäten in Bezug zu einem Ort. Die Entität „Objekttyp” erlaubt es, Aktionen auf bestimmte Objekte einzuschränken. So kann die Einschenkaktion nur auf Objekte vom Typ Glas bzw. Flasche angewandt werden. Die Symbole der Klassen „Ressource” und „Objekttyp” sind nicht von der Szene abhängig. Ein Symbol der Klasse „Objekt” beschreibt ein konkretes Exemplar eines Objektes in der Szene. Analoges gilt für „Ort”.

Die Symbole zur Beschreibung von Relationen sind in Tabelle 5.2 aufgeführt. Relationen zur Typisierung ordnen Entitätssymbolen einer Symbolklasse zu. Sie sind notwendig um Aktionen auf sinnvolle Entitäten zu beschränken. Eine Transportaktionen darf etwa nur Objekte zwischen Orten bewegen. Diese Einschränkung der Anwendbarkeit ist nicht manipulationsspezifisch, daher beziehen sich die Symbole „Ist Manipulator”, „Ist Objekt”, „Ist Vom Typ” und „Ist Ort” auf keine der im vorigen Kapitel diskutierten Aspekte.

Der Block Objekteigenschaften deckt Relationen ab, die Eigenschaften einzelner Objekte, abgesehen von ihrem Ort beschreiben. Neben generischen Eigenschaften, die auf alle Objekte anwendbar sind, gibt es spezifische Eigenschaften bestimmter Objekte. „Wird Manipuliert” bildet die Tatsache ab, dass ein Objekt gerade Teil einer Manipulationshandlung ist. Wie in

Relation	Bedeutung	Aspekt
Typisierung		
Ist Manipulator (A)	Bei A handelt es sich um einen Manipulator	
Ist Objekt (A)	Bei A handelt es sich um ein Objekt	
Ist Typ (A)	A ist ein Objekttyp	
Ist Ort (A)	Bei A handelt es sich um einen Ort	
Objekteigenschaften		
Wird Manipuliert (A, B)	A ist gerade Teil einer Manipulationshandlung der Ressource B	13
Ist Beweglich (A)	Der Roboter kann den Ort von A verändern	4
Ist Vom Typ (A, B)	Objekt A ist vom Typ B	4
Objektspezifisch	Relationen, die einen Zustand, der spezifisch für ein Objekt bzw. einen Objekttypen ist beschreibenm z.B. „Ist Offen(A)" für eine Flasche	7, 15
Lokalisierung		
An Ort (A, B)	Das Objekt A befindet sich am Ort B	5,11,14
Ort Unbekannt (A)	Der Ort des Objektes A ist nicht bekannt; bekannt ist lediglich die Existenz des Objektes	
Hindernisse		
Ist Frei (A)	Der Ort A ist nicht durch ein Objekt belegt.	8,11,14
Benötigt Freiraum (A, B)	Die Aktion A benötigt Freiraum am Ort B	8,11
Mechanik		
Stützt(A, B)	Objekt A übt eine Kraft auf B aus, ohne die B seinen Ort nicht statisch halten kann	9,17
Stabil (A, B)	Wird A bewegt, so wird auch B bewegt, und zwar in einer kontrollierten Weise	9,17
Unstabil (A, B)	Wird A bewegt, so wird auch B bewegt, aber in einer unkontrollierten Weise	9,17
Roboterzustand		
Belegt Durch (A, B)	Die Ressource A des Roboters ist durch eine Aktion B belegt	10,12, 16
Ressource Frei (A)	Die Ressource ist frei	10,12,16

Tab. 5.2.: Überblick über die verwendeten Symbolklassen zur Darstellung von Relationen. Die Gruppe der Typisierung dient der Zuordnung von Symbolen zu Symbolklassen. Neben generischen Relationen existiert bei den Objekteigenschaften eine Untergruppe von spezifischen Relationen, die nur für bestimmte Objekte und Aktionen sinnvoll sind. Die Gruppe der Lokalisierung und Hindernisse beschreibt den Ort von Objekten und ist eng mit dem geometrischen Modell der Szene verbunden. Darüber hinaus gehen die Relationen aus dem Bereich Mechanik ein. Der Roboterzustand spielt vor allem für die Verteilung von Aktionen auf Ressourcen eine wichtige Rolle.

Abschnitt 4.3 dargelegt ist, ist das symbolische Modell nur für Zeitpunkte zwischen Aktionen sinnvoll definiert. Wird ein Objekt durch einen Arm manipuliert, so muss es vor Einflüssen durch andere Ressourcen geschützt werden. Dies geschieht mit diesem Symbol. Es deckt damit implizit den Aspekt Zustand anderer Objekte (Aspekt 13) ab. „Ist Beweglich" beschreibt die Tatsache, dass ein Objekt vom Roboter bewegt werden kann. Dies ist ebenfalls eine generische Vorbedingung, die für jedes Objekt sinnvoll definiert werden kann. Sie ist vor allem für Transportaktionen wichtig und deckt für diese Aspekt Anwendbarkeit (Aspekt 4) ab.

Viele Manipulationsaktionen sind objektspezifisch. Um die Kompatibilität einer Aktion zu einem Objekt darstellen zu können, wird die binäre Relation „Ist Vom Typ" eingeführt. Sie ordnet einem Objekt eine Typ-Entität zu und deckt damit den Anwendbarkeitsaspekt des Objekttyps (Aspekt 4) für objektspezifische Aktionen ab. Neben den generischen Objekteigenschaften können spezifische Objekte weitere Eigenschaften haben, die Aspekte ihres Zustandes beschreiben. So kann etwa ein Glas voll oder leer sein, eine Flasche offen oder geschlossen. Beides sind relevante Aspekte für eine Einschenkaktion. Um diese berücksichtigen zu können, werden weitere objektspezifische Relationen zugelassen, die den Aspekt der Anwendbarkeit (Aspekt 4) abdecken, sowie ebenfalls spezifische Effekte einer Aktion modellieren, wie für den Aspekt Zustand assoziierter Objekte (Aspekt 15) notwendig.

Das „An Ort" Symbol verknüpft Objekte mit dem Ort, an denen sie sich zu einem Zeitpunkt befinden. Vom Ort hängt die Erreichbarkeit eines Objektes ab (Aspekt 5), ebenso wie die Kollisionsfreiheit (Aspekt 11). Weiter erlaubt die Relation die Darstellung des Effektes von Transportaktionen (Zustand assoziierter Objekte, Aspekt 14). Das „Ort Unbekannt" Symbol beschreibt den Zustand, in dem die Pose eines Objektes nicht bekannt ist. Das Symbol vervollständigt die möglichen Ortsbeschreibungen für Objekte.

Das „Ist Frei" Symbol beschreibt Orte, die von keinem Objekt belegt sind. Es ist eng mit dem Freiraummodell verbunden und deckt als solches Anforderungen an assoziierte Orte (Aspekt 8) ab. Ebenso wie „An Ort" spielt es auch in den Aspekt Kollisionsfreiheit (Aspekt 11). „Benötigt Freiraum" verbindet die Vorbedingung einer Aktion mit dem Orten der Objekte bzw. den freien Orten und adressiert damit ebenfalls die Aspekte Zustand assoziierter Orte (Aspekt 8) und Kollisionsfreiheit (Aspekt11).

Der Bereich der Mechanik der Szene wird durch die Symbole „Stützt", „Stabil" und „Unstabil" beschrieben. Sie decken gemeinsam die Aspekte 9 und 17 ab, also die Zustände von Objekten, die nicht direkt von der Aktion beeinflusst werden. Effekte von Aktionen auf unbeteiligte Objekte sind meist nicht vorhersehbar und unerwünscht. Daher wird durch die „Trägt" Relation nicht der Effekt bestimmt, sondern lediglich ausgesagt, dass ein Effekt auftreten kann. Im Falle der „Stabil" Relation bewegt sich ein Objekt kontrolliert mit einem anderen, dies kann für Aktionen ausgenutzt werden. Im Falle der „Unstabil" Relation dagegen kann über das Ergebnis

keine sichere Aussage gemacht werden. „Unstabil" tritt zusammen mit einem unkontrollierten Fallen eines Objektes auf, was meist unerwünscht ist.

Die letzte Gruppe von Relationen in Tabelle 5.2 beschreibt den Zustand des Roboters. Sie sind für den Scheduling Aspekt der Planung relevant, also für die Verteilung von Aktionen auf die ausführenden Komponenten. „Belegt Durch" zeigt dabei die Belegung einer Ressource durch eine Aktion an, „Ressource Frei" dagegen, dass die Ressource frei ist. Diese Relationen decken die Aspekte 10,12 und 16 ab.

Vergleicht man Tabelle 5.1 und Tabelle 5.2 mit der Auflistung aus Abschnitt 5.1, so ergibt sich, dass jeder der relevanten Aspekte von mindesten einem Symbol adressiert wird.

5.3. Aktionsmodell zur Planung von Manipulationsaktionen

Während das im vorigen beschriebene symbolische Zustandsmodell einen starken Bezug zur Szene und deren Wahrnehmung bzw. Interpretation hat, stellt das Aktionsmodell die abstrakte Anbindung des Planers an die Ausführung dar. Die Aktionen bestimmen die Handlungsmöglichkeiten des Roboters. In diesem Abschnitt wird zuerst auf die allgemeine Darstellung von symbolischen Aktionen eingegangen. Im Anschluss wird eine hierarchische Einteilung von Aktionen in Schichten definiert, um darauf aufbauend konkrete Aktionen und ihre Eigenschaften auf den Schichten zu betrachten.

5.3.1. Darstellung von Aktionen

Die Darstellung von Aktionen in dieser Arbeit basiert auf den Grundlagen des STRIPS Planers. Dessen Aktionsmodellierung wurde in Abschnitt 2.2.1 vorgestellt. Die Modellierung in dieser Arbeit geht jedoch über die reine STRIPS Repräsentation hinaus. Dies ist der Berücksichtigung von bimanuellen Aktionen geschuldet, die auch parallel oder überlappend ausgeführt werden können. Es werden daher auch die ausführende Ressource sowie die zeitliche Ausdehnung der Aktionen berücksichtigt. Als Erweiterung der Darstellung einer Aktion aus Name, Vorbedingungen und Effekt, werden in dieser Arbeit die Eigenschaften

- Name
- Ressource
- Objekt
- Vorbedinung
- Anfangs-Effekt und

- End-Effekt (Nachbedinung)

berücksichtigt. *Name* ist ein eindeutiger Name des Aktionstyps. Er kann für den Menschen zur Darstellung der Aktion verwendet werden und erlaubt die Zuordnung zu Manipulationsplanern für geplante Aktionen. *Ressource* bezieht sich auf eine Entität vom Typ Ressource, die die Aktion ausführen kann. Die Ausführung kann sich abhängig von der zugeordneten Ressource unterscheiden, etwa durch die unterschiedlichen Arbeitsbereiche der Arme. Objektaktionen (vgl. nächster Abschnitt) benötigen keine Ressourcen. Das *Objekt* (bzw. die Objekte) einer Aktion ist eine Liste von Objekten, die von der Aktion gezielt manipuliert werden. *Vorbedingungen* sind als eine Liste von Prädikaten definiert, die erfüllt sein müssen bzw. nicht erfüllt sein dürfen. Nicht erwähnte Prädikate sind irrelevant. Die Vorbedingungen können sich insbesondere auf das Objekt und die Ressource beziehen. Es werden Existenz- und Allquantoren verwendet, um Aussagen über alle Entitäten einer Klasse machen zu können. Da ihre Auswertung für den Planer aufwändig ist, wird ihre Nutzung im Aktionsmodell soweit möglich vermieden.

Die letzten Punkte der Aufzählung sind die *Anfangs-* und *End-Effekte* der Aktion. Diese beschreiben wie im STRIPS Modell Prädikate, die durch die Ausführung der Aktion wahr oder falsch werden. Nicht aufgeführte Prädikate bleiben unverändert. Die Aufteilung in Anfangs- und End-Effekte wird aufgrund der zeitlichen Ausdehnung der Aktion vorgenommen. Typischerweise treten manche Effekte zu Beginn einer Aktion ein. Diese werden durch Anfangseffekte modelliert. Wird ein Objekt gegriffen, so verlässt es zu Beginn der Aktion seinen Ort. Das Abstellen am Ende der Aktion dagegen kann durch einen End-Effekt beschrieben werden. Diese Modellierung ermöglicht es, den Szenenzustand zu Beginn einer Aktion so anzupassen, dass weitere, parallel ausgeführte Aktionen mit der laufenden Aktion konsistent geplant werden können. Sie können ebenfalls verwendet werden, um ein Prädikat nur während einer Aktion gelten zu lassen.

5.3.2. Hierarchisierung

Die Ausführung eines Plan ist durch sequenzieren von atomaren Roboteraktionen bestimmt. Die Menge der möglichen atomaren Aktionen für einen Roboter ist im Allgemeinen bekannt. Es erscheint jedoch unvorteilhaft, sich ausschließlich auf die atomaren Aktionen zu beschränken.

Ein Grund dafür ist die Tatsache, dass ein Aktionsmodell auf dieser Basis für einen Menschen nicht sonderlich anschaulich ist, und es daher schwierig zu entwerfen ist. Stattdessen fassen Menschen Aktionen zu größeren Teilaufgaben zusammen. Dies wird auch bei hierarchischer Planung ausgenutzt (vgl. Abschnitt 2.2.2). In diesem Abschnitt wird daher ein mehrschichtiges

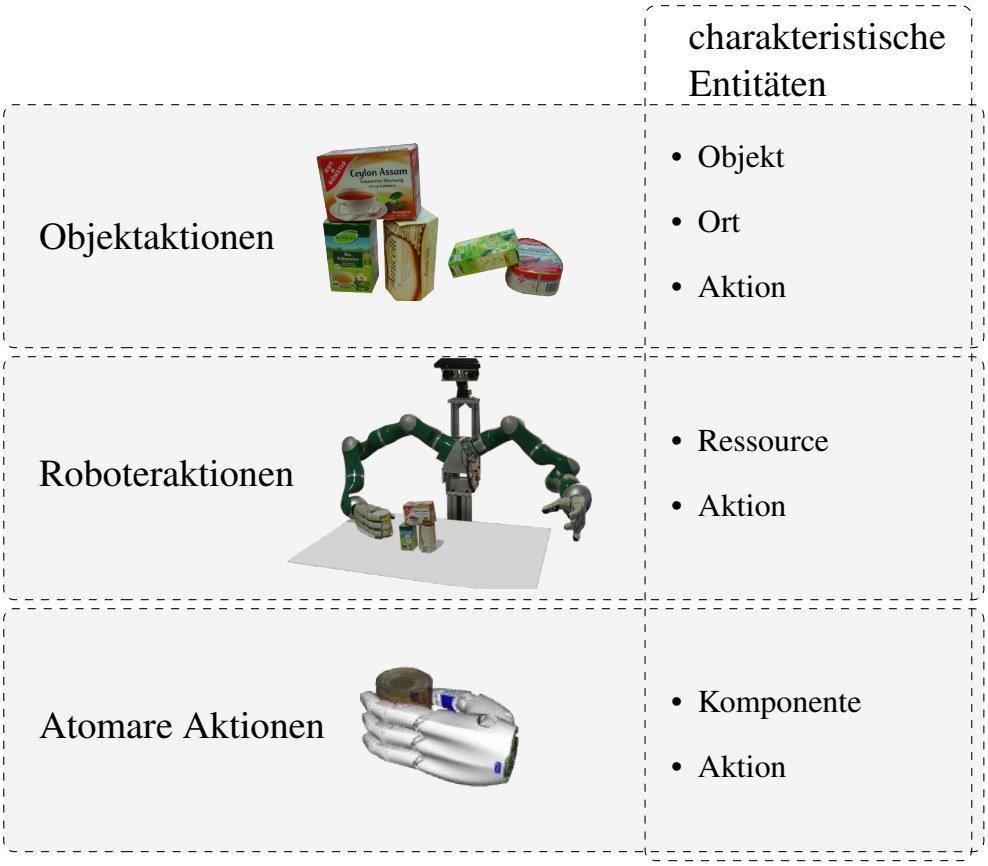


Abb. 5.1.: Verwendete Abstraktionsebenen des symbolischen Aktionsmodells. Auf der obersten *Objektaktionen*-Eben stehen die Objekte in der Szene und ihre Abhängigkeiten im Vordergrund. Auf der darunter liegenden *Roboteraktionen*-Ebene werden Roboteraktionen, die die Objekte in gewünschter Weise manipulieren, den Komponenten des Roboters zugewiesen. Die *Atomare Aktionen*-Ebene zerlegt diese in atomare Aktionen die vom Roboter ausgeführt werden können.

Aktionsmodell entwickelt, in dem die Schichten unterschiedlichen Abstraktionsebenen entsprechen.

Ein weiterer Grund, mit unterschiedlichen Abstraktionsebenen zu arbeiten ist die Performance der symbolischen Planung. Vorversuche mit einfachen Domänen haben gezeigt, dass symbolische Planung mit linearen Aktionsketten, die keine Hierarchisierung nutzen, von ihrer Performance nicht für die online Anwendung geeignet sind (Rühl u. a., 2013). In Versuchen mit dem in dieser Arbeit verwendeten Europa Planer mussten Anfragen mit Lösungen von mehr als zehn Aktionen Länge bei vielen Versuchen abgebrochen werden. Ohne Abstraktion entspricht das, je nach Domäne etwa zwei bewegten Objekten. Obwohl diese Arbeit keinen expliziten hierarchischen Planungsansatz verfolgt, können die Abstraktionsschichten im symbolischen Planer umgesetzt werden und tragen dort zur Steigerung der Performance bei.

Die symbolischen Aktionen werden in drei Schichten eingeteilt. Das Modell ist in Abb. 5.1 skizziert. Auf der obersten Schicht befinden sich die *Objektaktionen*. Diese basieren auf der

Beobachtung, dass auf einem hohen Abstraktionslevel Handlungssequenzen betrachtet werden können, ohne die ausführende Entität wie Mensch oder Roboter zu berücksichtigen. Die Aktionen sind hier nur durch ihre Auswirkung auf die Objekte beschrieben. Ebenso kommen die Vorbedingungen für die Aktionen aus der Szene selbst, etwa durch die mechanischen Relationen. Eine Aktion bzw. ein Plan auf dieser Ebene beschreibt demnach, wie sich Ort und Zustand der Objekte ändert. Dies ist ein ähnlicher Abstraktionslevel wie er in der Block Welt verwendet wird.

In der *Roboteraktionen*-Schicht werden die *Ressourcen* berücksichtigt. Sie sind physische Entitäten, die Objektaktionen in der Wirklichkeit umsetzen können. Bei der Handlungsplanung für Serviceroboter handelt es sich dabei typischerweise um Roboterarm und Greifer, die für eine einarmige Manipulationsaktion benötigt werden, oder um jeweils zwei für bimanuelle Aktionen. Dadurch kommen die Aspekte der Aktionsverteilung auf die ausführenden Komponenten hinzu, sowie die kinematischen Aspekte, die außerhalb des symbolischen Planers berechnet werden. Ebenso muss hier die zeitliche Ausdehnung der Aktionen berücksichtigt werden.

Auf der untersten Ebene, der *Atomare Aktionen*-Ebene spielen Objekte und Szene keine Rolle mehr, dort werden die Roboteraktionen auf atomare Aktionen abgebildet, die durch den Roboter ausgeführt werden können. So kann etwa ein Pick-and-Place in die atomaren Aktionen „Anrücken", „Hand-Öffnen" usw. zerlegt werden. Die erzeugten Aktionen werden von Manipulationsplanern auf Trajektorien abgebildet. Objekte und andere Symbole, auf die sich die Aktionen der höheren Schichten beziehen, müssen zu Parametern für die Manipulationsplaner abgebildet werden. Auch werden weitere atomare Aktionen eingeplant, die für den Betrieb des Roboters nötig sind, jedoch nicht aus der Aufgabe folgen. So kann die Objektlokalisierung während der Manipulation deaktiviert oder Parameter für die Ausführungsüberwachung gesetzt werden. Diese Ebene kapselt damit Implementierungsdetails des ausführenden Roboters von der Planung.

5.3.3. Beispiele modellierter Aktionen

Zwei konkrete Aktionen werden im Folgenden in Bezug auf ihre Modellierung genauer untersucht. Zum einen eine Transportaktion für ein Objekt, die durch den Roboter als Pick-and-Place Aktion umgesetzt wird. Sie ist durch ein umfangreiches symbolisches Modell beschrieben. Zum anderen wird die Einschenkaktion als eine allgemeine Manipulationsaktion beschrieben. Im Gegensatz zu Pick-and-Place steht über diese nur wenig Wissen zur Verfügung, das aus einem Manipulationsplaner gewonnen wird.

Name Transport	Ressource	Entitäten A, Ort, Ziel, Träger	Schicht Objektaktionenschicht
Vorbedingungen	Ist Objekt(A), Ist Ort(Ort), Ist Ort(Ziel), An Ort(A, Ort), Ist Frei(Ziel), Ist Beweglich(A), $\forall x \in \{\text{Träger}\}$: Stützt(x, A), $\forall x \notin \{\text{Träger}\}$: ¬Stützt(x, A) , $\forall x : \neg$Stützt(A, x)		
Anfangseffekte	Ist Frei(Ort), Wird Manipuliert(A), ¬ An Ort(A, Ort), Transport(A, Ort, Ziel), $\forall x : \neg$Stützt(x, A)		
Endeffekte	An Ort(A, Ziel), ¬ Ist Frei(Ziel), ¬ Wird Manipuliert(A) ¬ Transport(A, Ort, Ziel)		

Tab. 5.3.: Tranportaktion auf der Objektaktionenschicht

Name Pick-and-Place	Ressource B frei vom Typ Arm	Entitäten A, Ort, Ziel	Schicht Roboteraktionenschicht
Vorbedingungen	Ist Objekt(A), Ist Manipulator(B), Transport(A, Ort, Ziel), Ressource Frei(B)		
Anfangseffekte	¬Ressource Frei (B), Belegt Durch(B, A), Pick-and-Place(A, B, Ort, Ziel)		
Endeffekte	Ressource Frei(B), ¬Belegt Durch(B, A), ¬Pick-and-Place(A, B, Ort, Ziel)		

Tab. 5.4.: Die Pick-and-Place Aktion auf der Roboteraktionenschicht

Name Pick	Ressource B	Entitäten A, Ort, Ziel	Schicht Atomare Aktionen
Vorbedingungen	Pick-and-Place(A, B, Ort, Ziel), Anrücken Fertig(A, B, Ort, Ziel), ¬ Pick Fertig(A, B, Ort, Ziel)		
Anfangseffekte			
Endeffekte	Pick Fertig(A, B, Ort, Ziel)		

Tab. 5.5.: Die atomare „Pick” Aktion

Name Move	Ressource B	Entitäten A, Ort, Ziel	Schicht Atomare Aktionen
Vorbedingungen	Pick-and-Place(A, B, Ort, Ziel), Pick Fertig(A, B, Ort, Ziel), ¬ Move Fertig(A, B, Ort, Ziel)		
Anfangseffekte			
Endeffekte	Move Fertig(A, B, Ort, Ziel)		

Tab. 5.6.: Die atomare „Move” Aktion

Name Place	Ressource B	Entitäten A, Ort, Ziel	Schicht Atomare Aktionen
Vorbedingungen	Pick-and-Place(A, B, Ort, Ziel), Move Fertig(A, B, Ort, Ziel), ¬ Place Fertig(A, B, Ort, Ziel)		
Anfangseffekte			
Endeffekte	Place Fertig(A, B, Ort, Ziel)		

Tab. 5.7.: Die atomare „Place” Aktion

5.3.4. Transportaktion

Die Transportaktion dient dazu, ein Objekt von einem Ort zu einem anderen zu bewegen. Sie benötigt einen hohen Grad der Modellierungsgenauigkeit, damit sie vom Planer autonom zielführend eingesetzt werden kann.

Die Modellierung der Transportaktion auf der Objektaktionenschicht ist in Tabelle 5.3 dargestellt. Ressourcen werden auf dieser Ebene vernachlässigt. Die Aktion operiert auf mehreren Entitäten (genauer: Objekte, der Begriff wird in diesem Abschnitt jedoch vermieden, um Verwechselungen mit den physischen Objekten der Szene zu vermeiden). Die Variable A bezeichnet das transportierte Objekt; Ort und Ziel die Orte zwischen denen es bewegt wird. {Träger} beschreibt die Menge der Objekte, von denen A getragen wird.

Die ersten Vorbedingungen stellen sicher, dass die verwendeten Symbole Entitäten eines geeigneten Typs darstellen: Ist Objekt(A), Ist Ort(Ort), Ist Ort(Ziel). Die weiteren Prädikate beschreiben die Semantik der Transportaktion. Das Symbol Ort muss den aktuellen Ort des Objektes A beschreiben: An Ort(A, Ort). Damit ist sichergestellt, dass die Pose des Objektes bekannt ist. Ist Frei(Ziel) stellt sicher, dass der Zielort nicht von einem anderen Objekt belegt ist. Ist Beweglich(A) ist eine weitere, notwendige Vorbedingung. Die folgenden, mit Allquantoren beschriebenen Prädikate $\forall x \in \{\text{Träger}\} : \text{Stützt}(x, A)$, $\forall x \notin \{\text{Träger}\} : \neg\text{Stützt}(x, A)$ stellen sicher, dass die Menge {Träger} genau die Objekte enthält, die die Lage von A stützen. Als letzte Bedingung darf es kein Objekt geben, das von A gestützt wird: $\forall x : \neg\text{Stützt}(A, x)$. Damit werden unkontrollierte Effekt auf unbeteiligte Objekte vermieden.

Die Anfangseffekte der Aktion sind: Der ursprüngliche Ort wird frei (Ist Frei(Ort)), das Objekt befindet sich nicht mehr dort ($\neg$ An Ort(A, Ort)), es wird manipuliert (Wird Manipuliert(A)). Dadurch werden die „Stützt" Relationen der tragenden Objekte aus {Träger} aufgehoben ($\forall x : \neg\text{Stützt}(x, A)$). Als letztes wird ein Prädikat, das die Aktion darstellt, hinzugefügt (Transport(A, Ort, Ziel)). Dies ermöglicht die Abbildung der beteiligten Symbole auf die Robterabstraktionsschicht. Entgegengesetzt dazu modellieren die Endeffekte den Zustand des Zielortes nach dem Ablegen des Objektes, sowie den nun beendeten Manipulationszustand des Objektes.

Auf der Roboteraktionsschicht wird die Transportaktion durch eine Pick-and-Place Aktion (Tabelle 5.4) ersetzt. Aus dem erzeugten Transport Symbol der Transportaktion sind Objekt, Ort und Ziel vorgegeben. Auf dieser Ebene wird entschieden, welche Ressource, in diesem Fall welcher Arm, die Aktion ausführt. Dazu wählt der Planer eine geeignete Ressource B. B muss dazu die Vorbedingungen erfüllen, insbesondere Ressource Frei(B). Die Anfangs- und Endeffekt der Aktion modellieren die Ressourcenbelegung von B durch A und stellen weiter ein Pick-and-Place(A,B, Ort, Ziel) Symbol für die nächste Ebene bereit.

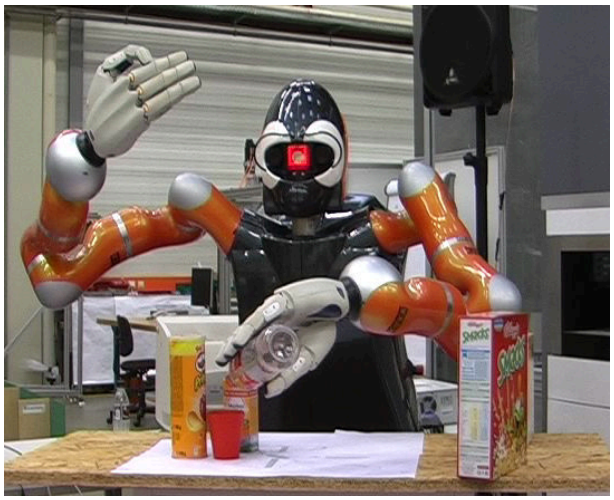


Abb. 5.2.: Ausführung der Einschenkaktion auf dem bimanuellen Demonstrator des Desire Projektes. Die rote Schachtel wurde vom Roboter beiseite geräumt, um die durch einen Manipulationsplaner erstellte Aktion ausführen zu können.

Auch die Anbindung an den kontinuierlichen Manipulationsplaner findet auf der Roboteraktionsschicht statt. Posen der Start- und Zielorte des Objektes können aus den Annotationen der Ortssymbole gewonnen werden, der Objekttyp aus dem Objektsymbol. Findet der Manipulationsplaner keine Trajektorie, die die Aktion im Kontinuierlichen beschreibt, so wird das Symbol verworfen.

Auf unterster Ebene werden die Aktionen in atomare Roboteraktionen zerlegt. Für die Transport bzw. Pick-and-Place Aktion muss eine Sequenz aus Anrücken, Pick (Tabelle 5.5), Move (Tabelle 5.6), Place (Tabelle 5.7) und Abrücken erstellt werden. Zur Parametrierung wird analog zur Roboteraktionsschicht auf das Pick-and-Place Symbol zurück gegriffen, das auf der höheren Ebene erstellt wurde. Jede atomare Aktion erzeugt als Endeffekt ein {Anrücken|Abrücken|Move|Place|Retreat}Fertig Symbol. Dessen Nichtexistenz ist eine der eigenen Vorbedingungen. Sie verhindert, dass die Aktion mehrfach geplant wird. Gleichzeitig ist das Fertig Symbol auch Vorbedingung der folgenden Aktion, wodurch die korrekte Reihenfolge der Aktionen erzwungen wird.

5.3.5. Geplante Manipulation: Einschenken

Die Einschenkaktion (Ausführung dargestellt in Abb. 5.2) ist ein Beispiel für eine durch einen Manipulatiosplaner geplante Aktion. Über diese Aktionen ist nur wenig spezifisches Wissen im symbolischen Modell enthalten. Als Effekte der Aktion im Symbolischen wird lediglich modelliert, dass die Objekte manipuliert werden (Tabelle 5.8). Andererseits ist die Aktion an spezifische Objekte A und B gebunden, sodass deren Typ in der Vorbedingung überprüft wird.

Name Transport	Ressource	Entitäten A, B, Ort-Flasche, Ort-Glas, Ort-1, Ort-2, ..., Ort-n	Schicht Objektaktionenschicht
Vorbedingungen	Ist Objekt(A), Ist Objekt(B), Ist Vom Typ(A, Glas), Ist Vom Typ(B, Flasche), Ist Ort(Ort), An Ort(B, Ort-Flasche), An Ort(A,Ort-Glas), Ist Frei (Ort_1), (Ort_2), ..., (Ort_n) $\forall x : \neg$Stützt(A, x), $\forall x : \neg$Stützt(B, x)		
Anfangseffekte	Wird Manipuliert(A), Wird Manipuliert(B)		
Endeffekte	$\neg$Wird Manipuliert(A), $\neg$Wird Manipuliert(B)		

Tab. 5.8.: Einschenkaktion auf der Objektaktionenschicht

Die Ortssymbole werden benötigt, um den Manipulationsplaner mit deren angehängten Posen zu parametrieren.

Die Symbole Ist Frei(Ort_1), Ist Frei(Ort_2), ..., Ist Frei(Ort_n) werden automatisch aus dem Manipulationsplanner für die Einschenaktion erzeugt. Details zur Erzeugung werden in Abschnitt 6.4 beschrieben. Je nach Szene und Aktion kann die Anzahl der Symbole variieren.

Aufgrund des eingeschränkten Wissens über die Aktion im Symbolischen erfolgt die Abbildung durch die Abstraktionsschichten des symbolischen Modells direkt durch Kopieren des Aktionssymbols. Auf der Roboteraktionsschicht werden die verwendeten Ressourcen als belegt markiert. Die Zuordnung zu den Ressourcen ist durch den Manipulationsplaner gegeben. Auf der atomaren Aktionsebene entspricht die Aktion genau einer atomaren Aktion, eben der durch den Manipulationsplaner erzeugten Trajektorie.

5.4. Symbolische Planung in EUROA-PSO

Um das vorgeschlagene Konzept und Modell zur symbolischen Planung umzusetzen, wird auf eine frei verfügbare Implementierung aus dem Stand der Technik zurück gegriffen. Entscheidend für die Auswahl des Systems ist dabei die Fähigkeit, Aktionen ausführenden Ressourcen zuteilen zu können. Klassische, im Kapitel „Grundlagen" vorgestellte Ansätze wie STRIPS (Fikes u. Nilsson, 1971) oder SHOP 2 (Nau u. a., 2003) scheiden damit aus. Stattdessen wird auf das Europa-PSO Framework (Frank u. Jonsson, 2002; Daley u. a., 2005; Bernardini u. Smith, 2007), das in Abschnitt 2.2.3 vorgestellt wurde, zurückgegriffen. Dabei handelt es sich um einen Constraint Satisfaction basierten Planer, dessen Domänenbeschreibungssprache NDDL (Abschnitt 2.1.5) das Konzept von Zeitstrahlen verwendet.

5.4.1. Darstellung des symbolischen Modells durch Zeitstrahlen und NDDL

Zeitstrahlen ermöglichen die Berücksichtigung von zeitlichen Aspekten bei der symbolischen Planung. So kann die Parallelität von Aktionen oder ihr gegenseitiges Ausschließen modelliert werden. Die in diesem Kapitel vorgestellten Symbole werden auf Zeitstrahlen abgebildet. Eine triviale Möglichkeit die Abbildung zu definieren, ist es jedem Symbol einen Zeitstrahl zuzuweisen, der die Prädikate „erfüllt" und „nicht erfüllt" aufweist. Dies entspricht offensichtlich der Anforderung, dass zu jedem Zeitpunkt genau ein Prädikat gilt. Im beschriebenen Modell gibt es Symbole, die sich mit Zeitstrahlen vorteilhafter modellieren lassen.

Ressourcen können als jeweils ein Zeitstrahl modelliert werden. Aktionen, die eine Ressource ausführt sind mögliche Prädikate. Damit unterscheiden sich Aktionen nicht von Zuständen. Es wird ein Prädikat „Keine" für die Ressourcen eingeführt, sodass für eine Ressource geplant werden kann, keine Aktion auszuführen. Auf der Ebene der Objektaktionen existieren keine ausführenden Ressourcen. Um die Unabhängigkeit der Aktionen zu erhalten, wird hier jedes Objekt als Zeitstrahl betrachtet, der eine Aktion ausführen kann.

Auch der Ort eines Objektes lässt sich auf einen Zeitstrahl abbilden, da sich ein Objekt zu jeder Zeit an genau einem Ort befindet. Zusätzlich wird das „Wird Manipuliert" Prädikat verwendet, wenn ein Objekt in der Hand gehalten wird. Das „Ort Unbekannt" Prädikat vervollständigt den Zeitstrahl. Ein weiterer Zeitstrahl wird für Orte eingeführt; dieser enthält das „Frei" Prädikat und das „Trägt" Prädikat. Damit besitzen das „An Ort" Prädikat des Objektes und das „Trägt" Prädikat des Ortes die selbe Semantik. Sie werden über über Zwangsbedingungen äquivalent gehalten. Analog werden „Stützt" und „Frei" behandelt, um mechanische Relationen zu beschreiben. Beispielhaft sind in n Abb. 5.3 einige Prädikate für eine Transport Aktion als Gantt Diagramm dargestellt.

Da es keine besondere Behandlung für Aktionen gibt, müssen deren Vorbedingungen sowie Anfangs- und Endeffekte durch Allen-Relationen (Abschnitt 2.1.5) erzwungen werden. Welche Relation verwendet wird, ist abhängig davon, ob der Zustand durch die Aktion verändert wird. Wird eine Vorbedingung durch einen Anfangseffekt beendet, so muss die „trifft" Relation verwendet werden. Wird sie durch einen Endeffekt beendet, so trifft die „beendet" Relation zu. Bleibt das Prädikat bestehen, so ist die Aktion im Prädikat „enthalten". Da Allen Relationen für jede mögliche Abhängigkeit von Start- und Endzeitpunkt eines Prädikats definiert sind, lässt sich diese Schema für alle weiteren Bedienungen erweitern.

Aufgrund der definierten Zeitstrahlen des Szenen- und Aktionmodells werden aus Vorlagen regelbasiert NDDL Dateien erzeugt, die von EUROPA-PSO interpretiert werden können.

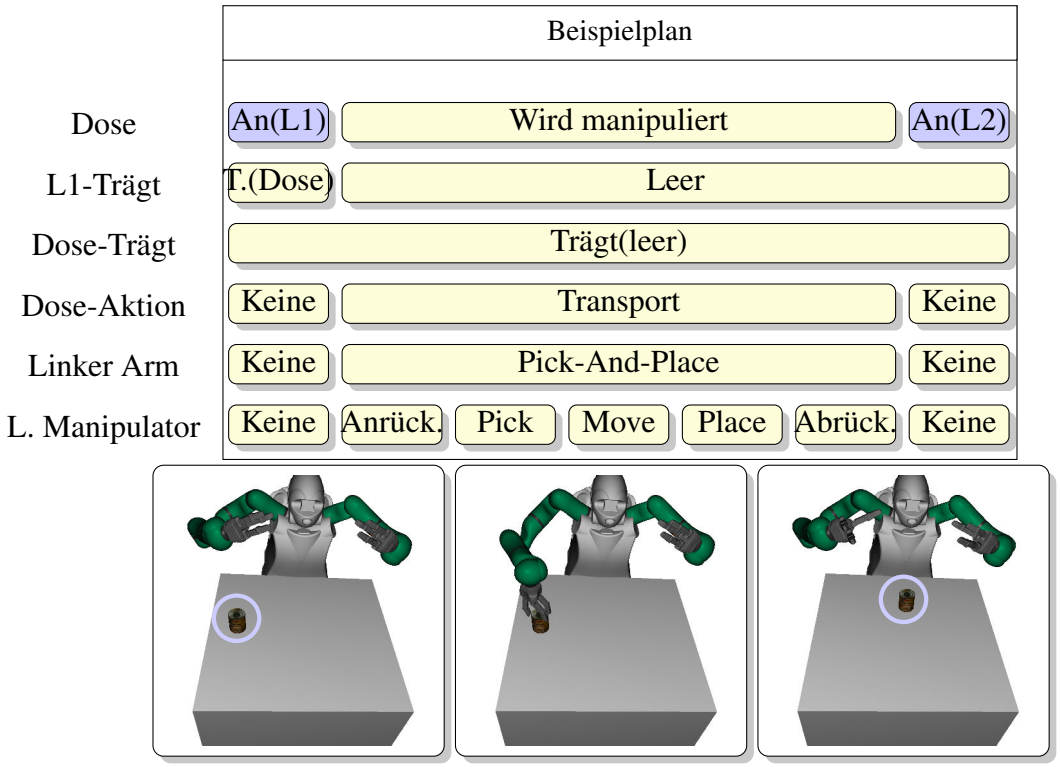


Abb. 5.3.: Gantt Diagramm eines Beispielplans für eine Pick-And-Place Aktion. Nur eine Teilmenge der Zeitstrahlen ist visualisiert.

In Abschnitt 5.3.2 wurde ein hierarchisches Aktionsmodell entwickelt. Der EUROPA-PSO Planer unterstützt von sich aus keine hierarchische Planung. Um das Modell in EUROPA-PSO umsetzen zu können, werden die Schichten von oben nach unten unabhängig voneinander geplant. Es wird zuerst die Domäne für die Objektaktionenebene geladen und ein Plan auf dieser Abstraktionsebene erstellt. Die Prädikate der Zeitstrahlen der Objektaktionen werden als Zwischenergebnisse zur Domäne der Roboteraktionen hinzugefügt, ebenso das ursprüngliche Szenen- und Objektmodell. Dies geschieht wieder regelbasiert aus Vorlagen in NDDL. Nach der Planung auf der Roboteraktionsebene wird der Prozess analog für die Ebene der atomaren Aktionen wiederholt, die atomaren Aktionen können aus den entsprechenden Zeitstrahlen extrahiert werden. Sie werden in Form von Flexiblen Programmen in einer Datenbank zur späteren Ausführung gespeichert.

Damit die beschriebene Form der Hierarchisierung funktionieren kann, muss das Ergebnis jeder Planungsebene mindestens eine real ausführbare Sequenz enthalten; ein Backtracking ist in dem Verfahren nicht möglich. Um dies in der Objektaktionsebene sicher zu stellen, werden hier Anfragen an die Manipulationsplaner gestellt, die sicherstellen, dass es eine kinematisch erreichbare Lösung auf der Roboteraktionsebene gibt.

5.4.2. Anbindung von Manipulationsplanern

Zur Anbindung von externen Programmroutinen bietet das EUROPA Planungssystem sogenannte *Constraints* an. Diese stellen aus Sicht des Planers ein Prädikat dar, das erfüllt sein muss, damit ein Token geplant werden kann. Sein Wahrheitsgehalt wird vom Framework durch den Aufruf einer Methode einer C++ Klasse bestimmt. Die Methode hat Zugriff auf weitere, aus dem Token heraus bestimmte Variablen, sowie die eine symbolische Repräsentation des Plans bzw. des symbolischen Modells. Mittels spezifischer, für das Constraint definierter Variablen werden wesentliche Eigenschaften wie Objekt ID und und Ort des Objektes an die Methode übergeben.

Die Orte von Hindernissen können aus dem Objektmodell ausgelesen werden. Dies wird zur Planung des Greifens und Ablegens eines Objektes genutzt. Aufgrund der diskreten Eigenschaften des symbolischen Zustandmodells gibt es nur endlich viele mögliche Konfigurationen der Szene. Da der Planer auch zeitliche Eigenschaften des Plans durch besuchen der Zustände plant, wiederholen sich die Anfragen an den Manipulationsplaner. Es wird ein Cache zwischengeschaltet, der innerhalb eines Planungsvorgangs bereits gestellte Anfragen an den Manipulationsplaner zwischenspeichert. Eine weitere Beschleunigung ergibt sich, da die Aktionen Greifen von A, Ablegen auf B und der umgekehrte Vorgang Greifen von B, Ablegen auf A als gleich behandelt werden. Bei der Greifplanung werden weiter nur Kinematik und Kollisionsfreiheit beim Greifen bzw. Ablegen berücksichtigt. Die kollisionsfreie Bahnplanung zwischen den Konfigurationen wird an dieser Stelle nicht berücksichtigt. In den Tischszenarien findet diese zwischen zwei erreichbaren Konfigurationen praktisch immer eine Lösung; der durch das Weglassen erzeugte Fehler fällt nicht ins Gewicht.

5.4.3. Pläne optimaler Länge

Der in EUROPA-PSO enthaltene Solver benutzt eine Tiefensuche zur Erzeugung eines Plans. Dies hat zwei wesentliche Nachteile: Da prinzipiell unendlich lange Pläne erzeugt werden können, muss die Suche nicht terminieren, sie kann in einem unendlich tiefen Zweig ohne Lösung gefangen sein. Zum anderen ist eine gefundene Lösung nicht optimal, es könnte sehr viel einfachere Lösungen geben, wenn eine frühe Entscheidung der Suche anders getroffen würde.

Diese Nachteile können durch Verwendung einer Breitensuche umgangen werden. Diese wird über die maximale Länge des erzeugten Plans aus der Tiefensuche erzeugt. Das Verfahren wird als iterative Tiefensuche (Russell u. Norvig, 2003) bezeichnet. Als weiterer Vorteil muss es, im Gegensatz zur Breitensuche, nicht Buch führen, welche Zustände es bereits besucht hat. Es ist damit sehr speicherplatzeffizient. Die maximale Länge des Plans wird bei dem Verfahren zuerst mit einer vom Ziel abhängigen Länge initialisiert, in dieser Arbeit wird die minimale Anzahl

der zu bewegenden Objekte verwendet. Wird keine Lösung gefunden, so wird die maximale Planlänge sukzessive erhöht, bis entweder ein Plan gefunden wurde, oder eine maximale Dauer der Suche überschritten wurde. Da die Größe des Suchbaumes und damit der zeitliche Aufwand der Suche mit der maximalen Planlänge exponentiell wächst, dominiert der letzte Schritt den Aufwand der Suche. Der Mehraufwand im Vergleich zu einer echten Breitensuche ist vernachlässigbar.

5.4.4. Zielzustand

Um den Planer anwenden zu können muss ein Zielzustand spezifiziert werden. In der Umsetzung besteht dieser aus Tokens, die im letzten Zeitschritt des Plans aktiv sind. Dabei werden nur relevante Tokens vorgegeben, andere Zeitstrahlen können undefiniert sein, sodass sie vom Planer konsistent gefüllt werden.

Um eine Szene für die Ausführung einer Aktion anzupassen, müssen deren Vorbedingungen in der Szene erfüllt werden. Die konkret zu erzeugenden Tokens sind gerade die Vorbedingungen der Aktion in der gegebenen Szene.

5.5. Fazit

In diesem Kapitel wurde das symbolische Modell und die Planung darauf behandelt. Die spezifischen relevanten Aspekte von Manipulationshandlungen wurden betrachtet und aufgrund dieser ein symbolisches Modell der Szene entworfen. Dieses kann neben geometrischen Aspekten wie dem Ort von Objekten auch mechanische Zusammenhänge abbilden, die beschreiben, wie sich die Bewegung eines Objektes auf andere Objekte auswirkt.

Das Aktionsmodell beschreibt die dem System zur Verfügung stehenden Aktionen. Das System kann zum einen a Priori bekannte Aktionen berücksichtigen, deren symbolische Beschreibung detailliert durch Experten modelliert ist. Zum anderen können generische Aktionen verwendet werden, deren symbolische Beschreibung in Form von Freiraumanforderungen aus den von autonomen Manipulationsplaneren erzeugten Trajektorien extrahiert wird. Eine hierarchische Zerlegung des Aktionsmodells erlaubt effizientes Planen.

Das symbolische Modell wurde im Plaungsframework EUROPA-PSO umgesetzt. Dazu wurde es auf die in NDDL verwendete Zeitstrahldarstellung abgebildet.

6. Symbolisierung von Umwelt und Aktionen

In diesem Kapitel wird der Vorgang der Symbolisierung beschrieben, also die Abbildung von Aspekten aus dem kontinuierlichen Modell auf Symbole des symbolischen Modells.

Wie in Kap. 5 beschrieben, hängen wesentliche Eigenschaften der Manipulation von der aktuellen Szene ab. Sie werden im symbolischen Modell zur Planung berücksichtigt. Die aktuelle Szene ist dem Roboter jedoch erst zur Ausführungszeit bekannt. Daraus ergeben sich zwei wesentliche Anforderungen an das Verfahren zur Erzeugung der symbolischen Beschreibung der Szene aus der Wahrnehmung:

- Autonomie
- Online Fähigkeit

Der Roboter muss in der Lage sein, zur Ausführungszeit das Modell seiner Umgebung autonom aufzubauen. Von Experten modelliertes Wissen kann für statische Aspekte verwendet werden. Aspekte die von der Szene abhängen, müssen online erzeugt werden. Dies muss in einem vertretbaren Zeitrahmen geschehen, da das Aufbauen des Modells die Ausführung der Handlung verzögert. Lange Wartezeiten vor der Ausführung einer Handlung sind dem Benutzer gegenüber nicht zu vertreten. Sie beeinflussen weiter die Wirtschaftlichkeit des Einsatzes eines Serviceroboters negativ und es besteht die Gefahr, dass sich die Umgebung während einer langen Planungsphase verändert.

In Abschnitt 6.1 wird ein Prozess vorgestellt, der die aufgeführten Anforderungen erfüllen kann. Dazu greift er auf Symbolisierungsoperatoren zurück, die wiederum einzelne Aspekte der Szene abbilden und für die die selben Anforderungen gelten. Die Implementierung der Symbolisierungsoperatoren ist in Abschnitt 6.2 bis Abschnitt 6.5 beschrieben.

6.1. Prozess der Symbolisierung

Die Symbole des Szenenmodells aus Abschnitt 5.2 lassen sich in zwei Klassen einteilen:

Szenenabhängig Szenenabhängige Symbole werden durch die aktuelle Szene definiert; sie müssen aus dieser geschlossen werden.

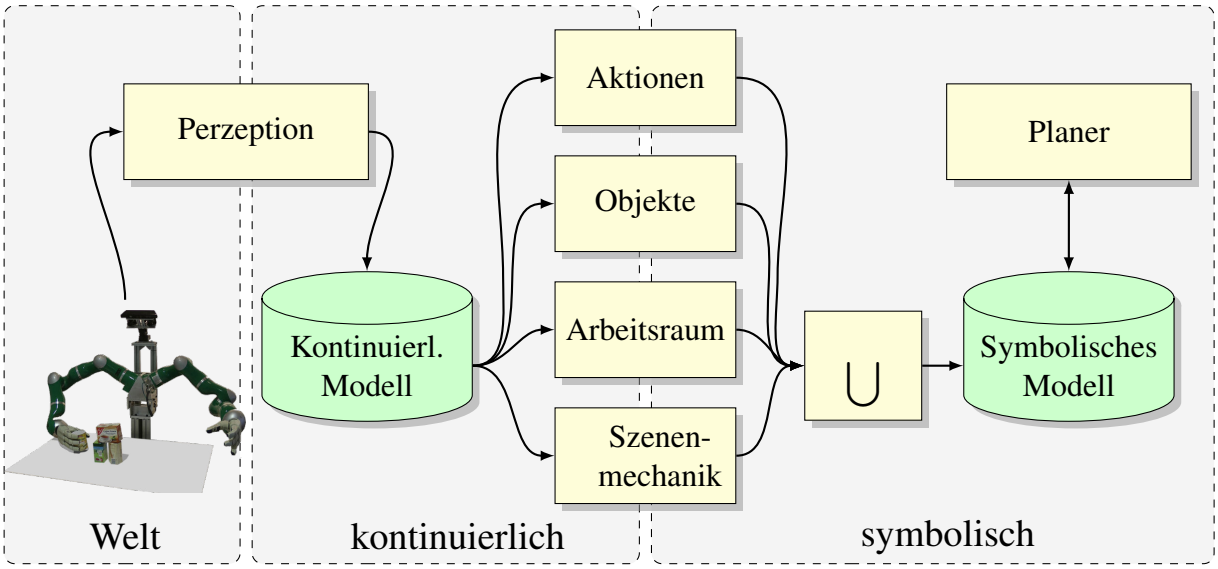


Abb. 6.1.: Überblick über den vorgeschlagenen Prozess der Symbolisierung. Die Eingabe der Symbolisierung ist das Wissen des kontinuierlichen Modells. Es beinhaltet insbesondere Wissen, welches aus der Szene durch die Perzeption gewonnen wird. Darüber hinaus kann es auch statisches Weltwissen enthalten. *Symbolisierungsoperatoren* bilden Aspekte des kontinuierlichen Modells auf Symbole ab. Die vier wichtigsten sind im Bild dargestellt: „Aktionen" werden durch ihre Freiraumanforderungen dargestellt, „Objekte" werden auf Objektsymbole sowie dazugehörige Ortssymbole abgebildet. Der „Arbeitsraum"-Operator erzeugt Ortssymbole für erreichbare Orte, an denen Objekte abgestellt werden können. „Szenenmechanik" erzeugt die symbolische Beschreibung der Abhängigkeiten der Objekte in der Szene in Bezug auf die Stabilität ihrer Lage. Die erzeugten Symbole werden zu einer Menge vereinigt und in das Szenenmodell eingefügt. Aufgrund des Szenenmodells erzeugt der symbolischer Planer einen Handlungsplan.

Vorgegeben Vorgegebene Symbole hängen von nicht von der aktuellen Szene ab, sondern von Objekten und Aktionen. Sie können vor der Ausführung festgelegt werden.

Sowohl szenenabhängige als auch vorgegebene Symbole können durch den Planer über die Zeit fortgeschrieben werden. Z.Bsp. gewinnen die Symbole der Belegtheitszustände von Ressourcen ihre Bedeutung nicht aus den Initialzuständen, sondern dadurch, dass der Planer damit einen Zustand kennzeichnet. Dem gegenüber stehen die Symbole der Szenenmechanik, die durch ihren szenenabhängigen initialen Zustand die Möglichkeiten der Handlung beeinflussen.

In diesem Abschnitt wird ein Prozess beschrieben, der das kontinuierliche Modell einer Szene auf das symbolische abbildet. Es wird davon ausgegangen, dass die Perzeption schnell genug ausgeführt werden kann, um das kontinuierliche Modell ständig zu aktualisieren. Das kontinuierliche Modell ist dann ein Abbild der realen Welt im Jetzt. Im Gegensatz dazu ist die Symbolisierung selbst ein rechenintensiver Prozess, der nur bei Bedarf vor der Planung durchgeführt wird.

Abb. 6.2.: Visualisierung der geometrischen, texturierten Modelle von 100 typischen Haushaltsobjekten aus (Kasper u. a., 2012), die in dieser Arbeit verwendet wurden.

Wie in Abschnitt 5.2 dargestellt, gibt es unterschiedliche, voneinander unabhängige Symbole, die aufgrund der Szene erzeugt werden. Sollen spezielle Eigenschaften von Objekten für Aktionen berücksichtigt werden, so können abhängig von den Aktionen zu diesen weitere hinzukommen. Um dies einfach im Prozess berücksichtigen zu können, werden die *Symbolisierungsoperatoren* eingeführt. Ein Symbolisierungsoperator bildet das kontinuierliche Modell auf eine Menge Symbole ab. Ein Operator kann mehrere Symbole erzeugen. Die Ergebnisse der Symbolisierungsoperatoren werden zu einer Vereinigungsmenge zusammengefasst und in das symbolische Modell integriert. Der gesamte Prozess ist in Abb. 6.1 dargestellt.

Die Symbolisierungsoperatoren für szenenabhängige, generische Symbole sind in Abbildung Abb. 6.1 dargestellt. Sie werden in den folgenden Abschnitten genauer beschrieben.

6.2. Symbolisierung der Objekte

Objekte sind die Entitäten, die der Roboter manipuliert. Abb. 6.2 zeigt eine Auswahl von Objekten aus dem Haushaltsbereich, die in dieser Arbeit verwendet werden. Der *Objekt-Symbolisierungsoperator* erzeugt aus dem kontinuierlichen Modell Objektsymbole. Für jedes Objekt in der Szene wird genau ein Symbol erzeugt. Im kontinuierlichen Modell sind die Objekte bereits mit einem Identifikator versehen, der die Zugehörigkeit zu einem Objekttyp beschreibt. Die entsprechenden Objekttyp-Symbole sind im symbolischen Modell statisch vorhanden. Das Wissen, wie die Symbole einem Identifikator zugeordnet werden, wird im Operator gespeichert. Damit kann zu jedem Objekt eine „Ist Vom Typ" Relation erzeugt werden. Ebenso wird, falls zutreffend, die „Ist Beweglich" Relation erzeugt. Die „Wird Manipuliert"

Name	Typ	Abdeckung
Erzeugt		
Objekt	Entität	vollständig
Ort	Entität	teilweise
Ist Vom Typ	Relation	vollständig
Ist Beweglich	Relation	vollständig
Wird Manipuliert	Relation	vollständig
An Ort	Relation	vollständig
Verwendet		
Objekttyp	Entität	-

Tab. 6.1.: Symbole, die durch den „Objekt-Symbolisierungsoperator" erzeugt und verwendet werden. Ein vollständige Abdeckung bedeutet, dass sämtliche Symbole dieses Typs aus dem Operator stammen, teilweise Abdeckung, dass es noch weitere Operatoren gibt, die das Symbol erzeugen.

Relation gilt zu Beginn eines Plans für kein Objekt, im Falle einer Neuplanung aufgrund einer Abweichung muss sie aus dem unterbrochenen Plan geschlossen werden. Um die Verständlichkeit eines Plans für einen Benutzer zu erhöhen, wird das Objektsymbol mit einem Namen, basierend auf dem Objekttyp annotiert.

Für jedes Objekt wird ein „Ort" Symbol erzeugt, das die initiale Position des Objektes beschreibt. Es wird mit der 6D Pose des Objektes annotiert. Eine „An Ort" Relation verbindet das Objekt mit dem Ort.

In Tabelle 6.1 sind die erzeugten Symbole des Objekt-Symbolisierungsoperators zusammengefasst. Der Operator liefert die vollständige Menge der Objekt-Entitäten, sowie die generischen Relationen zur Beschreibung von Objekten. Zur Menge der Orte liefert er einen Beitrag, der vom Arbeitsraum-Symbolisierungsoperator im folgenden Abschnitt vervollständigt wird.

6.3. Symbolisierung des Arbeitsraums

Ein Roboter manipuliert in einer sechsdimensionalen Umgebung. Soll ein Objekt abgestellt werden, muss sich der Planer für eine sinnvolle Lage des Objektes, in der es für den Manipulator erreichbar ist, entscheiden. Der symbolische Planer selbst ist nicht in der Lage, aufgrund des kontinuierlichen Raumes eine Objektlage auszuwählen. Daher wird in diesem Abschnitt ein Verfahren vorgestellt, dass den kontinuierlichen Raum auf eine endliche Menge „Ort" Symbole abbildet und damit den „Arbeitsraum" Symbolisierungsoperator implementiert. Der Symbolisierungsoperator erzeugt ausschließlich Ortsymbole. Tabelle 6.2 enthält daher nur diese Entität.

Ein „Ort" Symbol ist annotiert mit einer sechdimensionalen Lage, mittels der es im Prozess der Konkretisierung wieder auf einen kontinuierlichen Ort abgebildet werden kann. Für ein

Name	Typ	Abdeckung
Erzeugt Ort	Entität	teilweise, aus „Objekte”
Verwendet -		

Tab. 6.2.: Symbole, die durch den „Arbeitsraum-Symbolisierungsoperator” erzeugt werden. Eine teilweise Abdeckung bedeutet, dass es weitere Operatoren gibt, die das Symbol erzeugen.

Ort-Symbol, welches der verwendeten Menge an Symbolen hinzugefügt wird gilt: Unter Vernachlässigung von Hindernissen ist ein Objekt an diesem Ort für den Roboter erreichbar oder zumindest mit großer Wahrscheinlichkeit erreichbar.

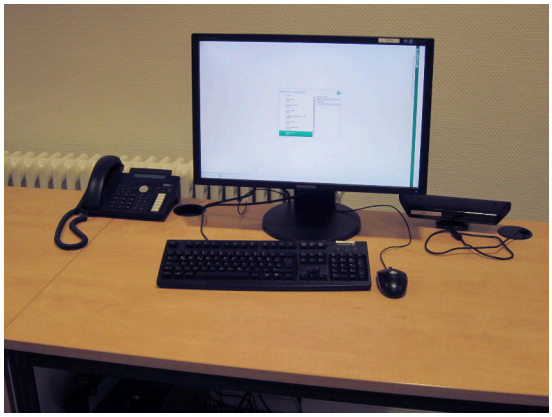
Im Folgenden wird zunächst in Abschnitt 6.3.1 die Tischannahme begründet, die von den Verfahren zur Symbolisierung des Arbeitsraums verwendet wird. Danach wird in Abschnitt 6.3.2 der Prozess zur Erzeugung von Ortssymbolen beschrieben. Abschnitt 6.3.3 beschreibt die zur Auswahl von Orten verwendete Bewertungsfunktion der *Greifbarkeit*.

6.3.1. Tischannahme

Beobachtet man Menschen bei Manipulationshandlungen, so erkennt man, dass Objekte die gerade nicht manipuliert werden, nicht an beliebigen Orten vorkommen. Ein Hauptgrund dafür ist die Schwerkraft, die es nicht erlaubt, Objekte im freien Raum zu platzieren; zumindest nicht statisch und kontrolliert, wie es für Manipulationsaufgaben gewünscht ist. Stattdessen werden weitere Objekte benötigt, welche die notwendige Gegenkraft aufbringen, um das Objekt an seinem Ort zu halten.

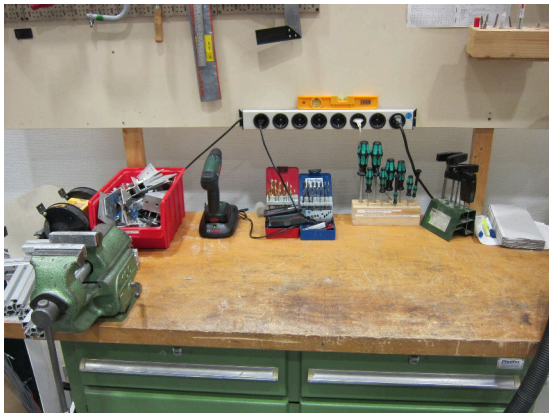
Ein Objekt, das ein anderes trägt, hat dazu meist auf der Oberseite eine horizontale ebene Fläche, das getragene Objekt entsprechend auf der Unterseite. Dadurch entsteht ein guter Halt auch wenn zwischen den Objekten nur wenig Reibung wirkt. Menschen verwenden meist Tische, um Objekte in einer ihnen angenehmen Höhe abzustellen. Beispiele für solche Tische sind in Abb. 6.3 dargestellt. Neben dem klassischen Schreibtisch (Abb. 6.3(a)) kommen weitere Ausprägungen vor, etwa eine Arbeitsfläche in einer Küche (Abb. 6.3(b)) oder eine Werkbank (Abb. 6.3(c)) in der Werkstatt. In dieser Arbeit wird angenommen, dass Manipulation grundsätzlich auf Tischen, also auf einer horizontalen Ebene stattfindet. Diese Annahme wird als *Tischannahme* bezeichnet.

Objekte können noch an weiteren Orten vorkommen. Mehrere Ebenen übereinander existieren z.B. in Schränken oder auf Regalen. Dort ist es möglich, die Ebenen als mehrere Tische anzusehen und damit die Tischannahme zu verwenden. Zwei weitere Gegenbeispiele können in Abb. 6.3 gefunden werden. In Abb. 6.3(c) ist ein Schraubstock zu sehen, in dem Objekte ein-



(a)

(b)



(c)

(d)

Abb. 6.3.: Fotos von Arbeitsräumen, in denen Menschen manipulieren. (a) zeigt einen Schreibtisch, 6.3(b) Arbeitsflächen in einer Küche. In (c) ist eine Arbeitsfläche einer Werkbank zu sehen, in (d) ein Rednerpult und ein Stehtisch in einem Präsentationsraum. Allen Szenarien sind die ebenen Flächen zum Abstellen von Objekten gemeinsam.

gespannt werden können. Ihre Lage verletzt die Tischannahme. Allerdings kann die Verletzung durch eine spezialisierte Aktion zum Einspannen umgangen werden. In Abb. 6.3(b) ist im Hintergrund ein Handtuch über einen Stab am Backofen gehängt. Es gilt dasselbe wie im Beispiel des Schraubstocks. Die grundsätzliche Annahme, dass Manipulation auf Ebenen stattfindet, wird durch die aufgeführten Beispiele nicht eingeschränkt.

6.3.2. Prozess zur Symbolisierung des Arbeitsraums

Der Prozess zur Symbolisierung des Arbeitsraums ist in Abb. 6.4 dargestellt. Er wird ausgehend vom sechsdimensionalen kontinuierlichen Raum bis zum Ergebnis des Symbolisierungsoperators, einer Menge von Ortsymbolen, in der Reihenfolge seiner Abarbeitung beschrieben. Die

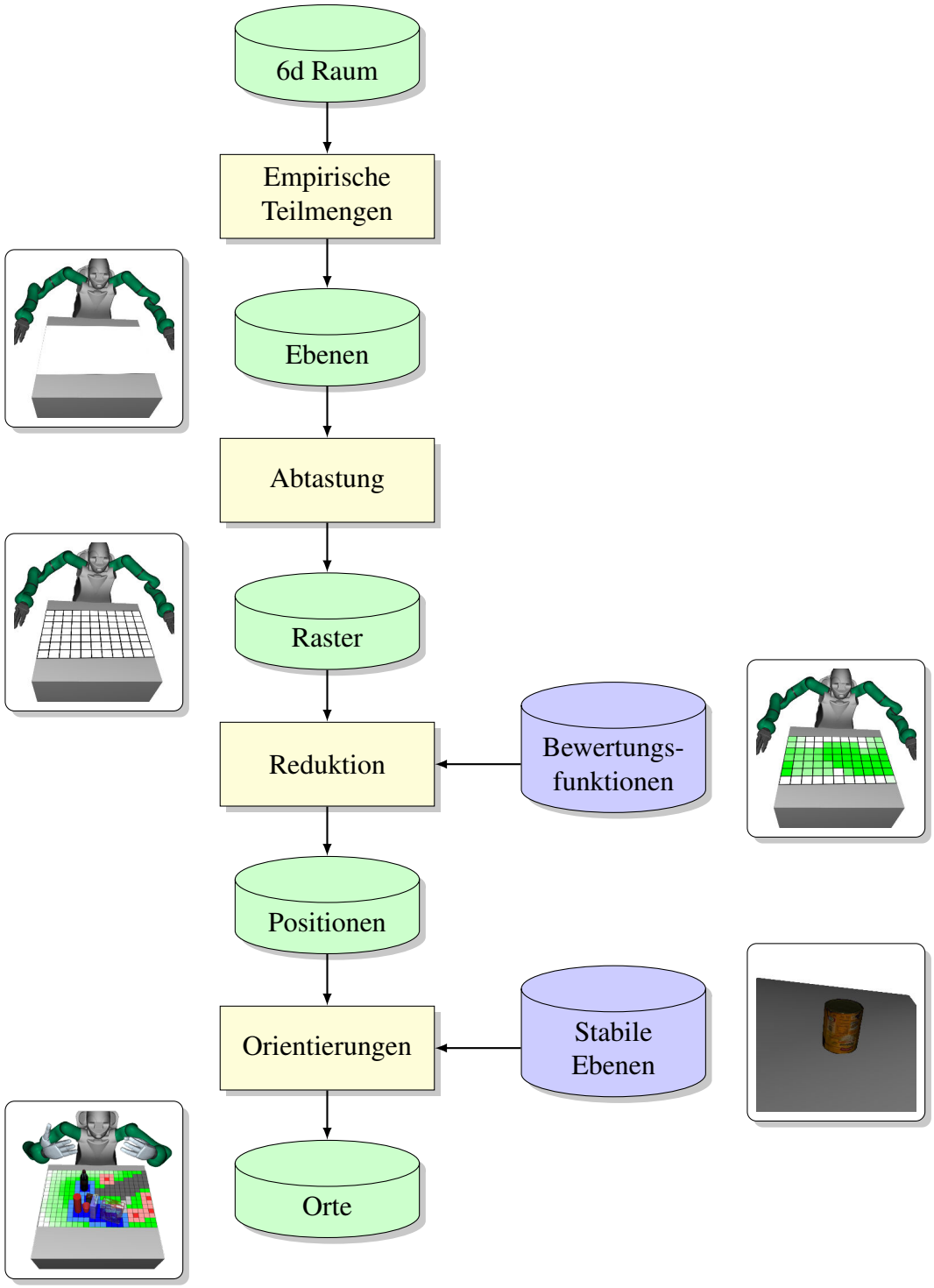


Abb. 6.4.: Prozess zur Symbolisierung des Arbeitsraums. Der kontinuierliche sechsdimensionale Raum wird auf empirische Teilmengen reduziert. Als empirische Teilmengen dienen Tische, auf denen Manipulationshandlungen durchgeführt werden. Der Tisch wird zu einem Raster abgetastet. Orte in dem Raster werden mittels einer Bewertungsfunktion bezüglich ihrer Eignung für Manipulationsaktionen bewertet. Aufgrund der Bewertung erfolgt eine Reduktion auf eine begrenze Menge Orte, die das Ergebnis des Symbolisierungsoperators darstellt und zur Planung verwendet wird.

erzeugte Menge von Ortssymbolen sollte einen geringen Umfang haben, um eine effiziente Planung zu ermöglichen.

Im ersten Schritt werden aus dem sechsdimensionalen Arbeitsraum empirisch bestimmte Teilmengen erzeugt. Diese basieren auf der Beobachtung, dass Objekte nicht an beliebigen Orten vorkommen. Die im vorigen Abschnitt vorgestellte Tischannahme besagt, dass ebene Flächen, wie sie auf Tischen vorkommen, für Manipulationsaufgaben eine besondere Bedeutung haben. Als empirische Teilmengen werden daher die ebenen Flächen in der Szene ausgewählt. In den räumlichen Dimensionen führt dies zu einer Reduktion auf zwei Dimensionen mit endlicher Ausdehnung in x und y Richtung. Diese sind weiterhin kontinuierlich. Die Z-Koordinate wird auf eine diskrete, endliche Menge an Werten begrenzt.

Der Abtastungsschritt des Verfahrens reduziert die erzeugten Ebenen auf eine diskrete, endliche Menge an Positionen im Raum. Die Orientierung bleibt weiterhin kontinuierlich. Es ergibt sich eine Menge von Orten, die in Rechteckrastern auf den Tischen verteilt sind.

Im Reduktionsschritt wird objektspezifisches Wissen verwendet, um die Menge der Orte zu reduzieren. Mittels einer Bewertungsfunktion werden die erzeugten Orte bezüglich ihrer Eignung zur Manipulation bewertet. Die verwendete Bewertungsfunktion der Greifbarkeit wird im nachfolgendem Abschnitt beschrieben. Die Bewertungsfunktion ist von den verwendeten Objekten abhängig. Das Wissen, welche Objekte in der Szene sind, ist an dieser Stelle vorhanden. Die erzeugten Orte werden mittels eines gierigen Algorithmus ausgedünnt. Dieser nutzt die Diskretisierung, um zu vermeiden, Orte in der Nachbarschaft von bereits betrachteten Orten zu verwenden. Dies verhindert das Überlappen von Orten, was unerwünscht ist, da es an einem freien Ort möglich sein muss, ein Objekt zu platzieren. Es führt ebenso zu einer Verteilung der erzeugten Orte über den Arbeitsraum. Der verwendete Algorithmus ist in Algorithmus 6.1 gezeigt. Das Ergebnis des Schrittes sind Positionen mit nicht spezifizierten Orientierungen. Diese Positionen werden weiter verarbeitet.

Im Orientierungs-Selektionsschritt werden die erzeugten Positionen mit den Orientierungen der stabilen Lagen der Objekte zu Orten ergänzt. Es werden alle Kombinationen aus Positionen und Orientierungen der stabilen Lagen erzeugt. In einem abschließenden Reduktionsschritt werden die Orte verworfen, für die keine der beiden folgenden Bedingungen erfüllt ist:

1. Es gibt einen Griff, mit dem das Objekt sowohl an seinem initialen Ort in der Szene als auch am erzeugten Ort gegriffen werden kann.
2. Es gibt einen Griff, mit dem das Objekt sowohl an seinem Zielort in der Szene als auch am erzeugten Ort gegriffen werden kann.

Algorithmus 6.1 Gieriger Algorithmus zur Erzeugung von Orten aus bewerteten Positionen

```
1:  function ERZEUGE ORTE(Orte aus Objektsymbolisierung O_objekte, Orte aus Prozess
    O_gen, Schwellwert τ, Bewertungsfunktion Bewertung, Gesuchte Anzahl n)
2:      R ← ∅
3:      for o ∈ O_objekte do
4:          R ← R ∪ {o}
5:          for t ∈ O_gen do
6:              if Distanz(t, o) < τ then
7:                  O_gen ← O_gen \ {t}
8:              end if
9:          end for
10:     end for
11:     while ‖R‖ < n ∧ O_gen ≠ ∅ do
12:         Wähle o ∈ O_gen mit Bewertung(o) maximal
13:         R ← R ∪ {o}
14:         for t ∈ O_gen do
15:             if Distanz(t, o) < τ then
16:                 O_gen ← O_gen \ {t}
17:             end if
18:         end for
19:     end while
20:     return R
21: end function
```

6.3.3. Greifbarkeit

Die Greifbarkeit ist ein Maß zur Bewertung eines Ortes bezüglich seiner Eignung zur Manipulation eine Objektes. Sie ist abhängig vom Objekt und vom Roboter, insbesondere von seiner Kinematik. Vom Objekt hängt sie über die Menge der möglichen Griffe ab. Im Folgenden wird die Greifbarkeit definiert und ihre praktische Berechnung hergeleitet.

Um die Greifbarkeit als objektabhängiges Maß als Bewertungsfunktion für den Prozess der Ortssymbolisierung einsetzen zu können, werden Ansätze zur Fusion vorgeschlagen.

6.3.3.1. Motivation und Definition

Aufgrund der Beobachtungen der Tischannahme aus Abschnitt 6.3.1 kann eine zweite, schwächere Annahme getroffen werden: In den meisten Fällen ist die Orientierung eines Objektes um die Z-Achse (Gierwinkel) für die Aufgabe irrelevant. Ist das nicht der Fall, so kann sie exakt spezifiziert werden. Der erste Fall tritt ein, wenn der Ort des Objektes lediglich die Aufgabe hat, das Objekt zu tragen. Dies ist für alle Orte, die zur temporären Ablage gewählt werden der Fall.

Die Annahme gilt nicht für die beiden weiteren Freiheitsgrade der Orientierung, den Roll- und Nickwinkel. Zur Plausibilisierung stelle man sich den Effekt auf ein Glas Wasser vor. Bedingt ist die Besonderheit des Gierwinkels durch die selben Vorbedingungen, die auch zur Tischannahme führen. Da sich Objekte in stabilen, tragenden Konstellationen im Allgemeinen in zwei Ebenen berühren, werden durch die parallelen Normalenvektoren der Ebenen die Ausrichtung der Objekte bis auf die Rotation um den Normalenvektor festgelegt. Aufgrund der Gravitation bzw., der Kraft die entgegen der Gravitation auf das getragene Objekt wirkt, ist der Normalenvektor parallel zur globalen Z-Achse. Eine Rotation um den Vektor ist möglich, da die Ebenen parallel bleiben und die Richtung der Gravitationswirkung unverändert bleibt.

Sei nun der Vektor $\mathbf{r}$ aus den kartesischen Koordinaten x,y,z und der Orientierung roll, pitch und yaw aufgebaut. Basierend auf der Annahme, dass der Gierwinkel beim Plazieren eines Objektes eine untergeordnete Rolle spielt, wird die Greifbarkeit $G_o(\mathbf{r})$ eines Objektes o an einer Pose $\mathbf{r}$ mit der folgenden Definition festgelegt.

Definition 6.1 ***Greifbarkeit***

$$
\begin{aligned}
g_o(\mathbf{r}) &= \begin{cases} 1 & \textit{falls o am Ort } \mathbf{r} \textit{ greifbar ist} \\ 0 & \textit{sonst} \end{cases} \\
\tau(\gamma) &= \begin{pmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{pmatrix} \\
G_o(\mathbf{r}) &= \frac{1}{2\pi}\int_0^{2\pi} g_o\left(\tau(\gamma) \circ T(\mathbf{r})\right) d\gamma
\end{aligned} \tag{6.1}
$$

In der Definition stellt $\tau(\gamma)$ die Rotation des Objektes um die Z-Achse dar. $T(\mathbf{r})$ bildet die sechdimensionale Pose des Objektes auf die die entsprechende Transformationsmatrix ab. Durch die Integration von γ über 2π in der Definition vom G_o wird die ursprüngliche Orientierung des Ortes $\mathbf{r}$ um die Z-Achse irrelevant, für jeden Gierwinkel ist das resultierende G_o gleich. Die Normierung mit $\frac{1}{2\pi}$ führt zu einem Wertebereich der Greifbarkeit von $[0,1]$.

In Worten kann die Greifbarkeit als der Anteil der möglichen Orientierungen um die Z-Achse eines Objektes an den gesamt möglichen Orientierungen um die Z-Achse des Objektes an einem Ort gedeutet werden.

Besondere Aufmerksamkeit gilt der Funktion g_0, welche entscheidet, ob ein Objekt o an einem Ort $\mathbf{r}$ greifbar ist. In dieser Funktion liegt die Abhängigkeit der Greifbarkeit vom Objekt bzw. seiner Geometrie begründet, ebenso wie vom Roboter bzw. Hand- und Arm-Kinematik und dem Ort des Roboters. Weiter hängt die Greifbarkeit eines Objektes auch von der umgebenden Szene ab, die für die Bestimmung genau spezifiziert wird. Aufgrund der Überlegungen aus

Abschnitt 6.3.1 wird eine leere Szene mit einem Roboter, dem Objekt und einem Tisch direkt unter dem Objekt festgelegt.

Es ist möglich, die Greifbarkeit mit einer kontinuierlich definierten Funktion g_o zu erweitern. Diese kann die Anzahl möglicher Griffe berücksichtigen, sowie deren Qualität. Eine weitere Erweiterung ist die Berücksichtigung der Genauigkeit der Kinematik an einem anzufahrenden Ort. g_0 beschreibt dann, ob die die resultierende Positioniergenauigkeit des Tool Center Point genügt, um einen Griff erfolgreich auszuführen. Die Erweiterungen werden im Rahmen dieser Arbeit nicht weiter verfolgt.

6.3.3.2. Diskretisierung

Zur Bestimmung von g_0 werden Methoden der Manipulationsplanung im Kontinuierlichen eingesetzt. Insbesondere werden Greifplanung und inverse Kinematik verwendet. Diese Algorithmen sind nicht kompatibel zu einer kontinuierlichen Integration über einen Wertebereich von γ, wie sie zur Definition der Greifbarkeit verwendet wurde. Für die praktische Bestimmung der Greifbarkeit müssen daher numerische Verfahren verwendet werden, die die Integration auf ein Abtasten des Parameters γ reduzieren.

Das Maß der Greifbarkeit soll die Struktur des Arbeitsraums des Roboters darstellen. Daher ist man nicht an einem einzelnen Wert der Greifbarkeit interessiert, sondern an einer Karte des gesamten Arbeitsraums. Um dies zu erreichen, werden die Objektpositionen $\mathbf{r}$ abgetastet. Man erhält daraus eine fünfdimensionale Karte. Eine solche Karte des Arbeitsraums ist aufwändig zu berechnen und zu speichern. Daher werden die Methoden, die bereits im Abschnitt 6.3.2 vorgestellt wurden, verwendet um die Dimensionalität zu reduzieren.

Basierend auf der Tischannahme wird die Greifbarkeit auf horizontale Ebenen beschränkt. Ausgehend von einer typischen Schreibtisch- und Arbeitsflächenhöhe für stehende Tätigkeiten, wird eine Arbeitsfläche in 80 cm Höhe als Basis verwendet. Von dort kann eine Abtastung in Z Richtung erfolgen. In Versuchen wurde feststellt, dass der Arbeitsraum in mehr als 30 cm vertikaler Entfernung für typische Manipulatorkonfigurationen bereits sehr eingeschränkt ist. Diese Arbeit fokussiert daher in Versuchen auf eine Ebene für z=80 cm.

Basierend auf der Tischannahme werden, wie in Abschnitt 6.3.2, die möglichen Orientierungen von Objekten auf solche reduziert, in denen das Objekt auf einer stabilen Ebenen zu stehen kommt. Bei den Objekten in Abb. 6.2 existieren durchschnittlich zwölf stabile Ebenen pro Objekt.

Es bleiben zwei kontinuierliche Parameter: die horizontale Position x und y. Diese werden innerhalb des Arbeitsbereiches abgetastet. Insgesamt werden drei translatorische Dimensionen

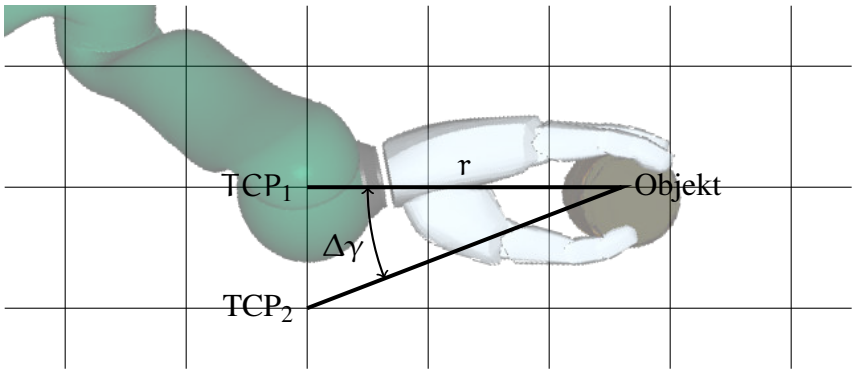


Abb. 6.5.: Die Diskretisierung des Gierwinkels ist durch die Diskretisierung der Ebene und den Abstand des Greifermittelpunktes vom TCP des Manipulators bestimmt.

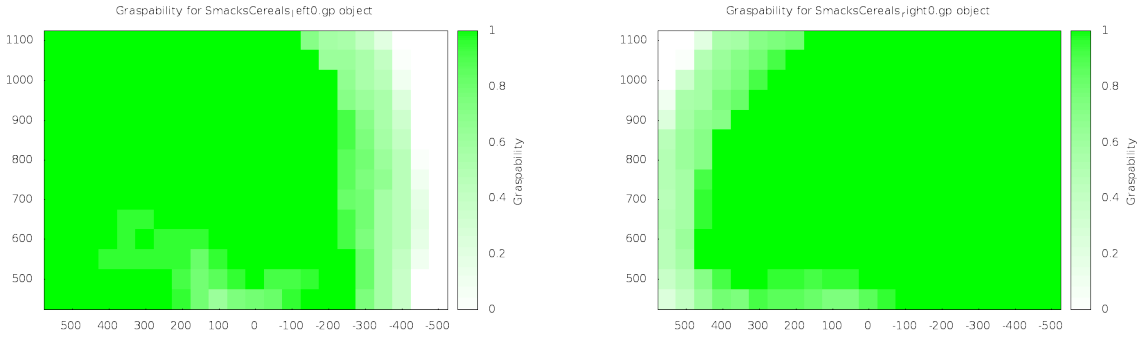
abgetastet sowie eine Orientierung. Zwei weitere Dimensionen der Orientierung werden aus einer diskreten Menge erzeugt.

Die Abtastrate $\Delta\gamma$ des Gierwinkels γ wird über den Abstand Δx zweier Punkte im kartesischen Raster bestimmt. Die kinematische Erreichbarkeit einer Objektpose durch einen Roboterarm hängt vom aus dem Griff resultierenden TCP ab. Wird das zu greifende Objekt rotiert, so hat dies aufgrund der Translation zwischen Objekt und TCP sowohl eine Translation als auch eine Rotation des TCP zur Folge. Δx wird nun so gewählt, dass die durch die Rotation entstehende Translation des TCP gerade einem Gitterabstand Δx entspricht. Dies ist in Abb. 6.5 visualisiert. Es ergibt sich für $\Delta\gamma$:

$$\Delta\gamma = \arcsin\left(\frac{\Delta x}{r}\right) \tag{6.2}$$

wobei r die Distanz zwischen gegriffenen Objekt und dem TCP des Manipulators ist. Die Distanz ist abhängig vom Griff. Praktisch ist die Varianz jedoch ($< \frac{r}{2}$) gering. Der Objektmittelpunkt kann durch einen empirisch ermittelten Punkt zwischen den Fingern bzw. Greiferbacken ersetzt werden. Für die Schunk Anthropomorphe Hand ergeben sich Werte für $r \approx 22$ cm, was zu $\gamma = 27^\circ$ bei $\Delta x = 10$ cm führt, oder $\gamma = 13^\circ$ für $\Delta x = 5$ cm. Dementsprechend sind 13 bzw. 27 Orientierungen zu berücksichtigen.

Im Diskreten ist die Greifbarkeitskarte wie folgt definiert:



(a) Linker Arm (b) Rechter Arm

Abb. 6.6.: Greifbarkeit des quaderförmigen „Smacks"-Objektes. In der Visualisierung stünde der Manipulator am unteren Bildrand.

Definition 6.2 (Greifbarkeitskarte)

$$
\begin{aligned}
\bar{G_o}(\mathbf{r},\omega) &= \frac{1}{\lfloor \frac{2\pi}{\Delta\gamma} \rfloor} \sum_{n=0}^{\lfloor \frac{2\pi}{\Delta\gamma} \rfloor} g_o\left(\tau(\omega, n\Delta\gamma) \circ T(\mathbf{r})\right) \qquad (6.3)\\
\mathbf{r} &= (x,y,z,0,0,0)\\
x &\in \{x_{min} + i \cdot \Delta x | i \in \mathbb{N}\} \cap [x_{min}, x_{max}]\\
y &\in \{y_{min} + i \cdot \Delta x | i \in \mathbb{N}\} \cap [y_{min}, y_{max}]\\
z &\in \{z_{min} + i \cdot \Delta x | i \in \mathbb{N}\} \cap [z_{min}, z_{max}]\\
\omega &\in \{\textit{Stabile Lagen von } o\}
\end{aligned}
$$

Die Greifbarkeitskarte hat damit drei Dimensionen und ist für Objekte und deren stabile Ebenen definiert.

Zur Berechnung der Greifbarkeit wird auf eine Griffdatenbank für die bekannten Objekte zurückgegriffen. Die Griffe der Griffdatenbank wurden mit dem Verfahren aus (Xue u. a., 2009, 2010) erzeugt. Es wurde in Abschnitt 2.4.2 beschrieben. Sowohl die zugrunde liegenden Objektmodelle als auch die Griffe sind in der KIT Objektdatenbank[1] (Kasper u. a., 2012) frei zugänglich.

6.3.3.3. Fusion

Für die Anwendung als Bewertungsfunktion zur Symbolisierung des Arbeitsraumes ist die diskrete Definition der Greifbarkeit $\bar{G}_o(\mathbf{r}, \omega)$ nicht ausreichend, da sie von den stabilen Lagen des Objektes abhängt. Auch ist sie nur für einen Manipulator definiert. Eine Möglichkeit ist es, dies bereits bei der Erzeugung zu berücksichtigen. So könnte man g_0 durch eine Funktion ersetzen, die Orte nur dann als greifbar bewertet, wenn dies für alle Lagen und Manipulatoren gilt. Dadurch würde allerdings Wissen über den Arbeitsraum verworfen, das später noch nutzbringend eingesetzt werden kann. So ist ein guter Ablageort für eine Übergabeaktion von einen Arm an den anderen eine andere, als für ein einfaches Verräumen eines Objektes. Der verwendete Ansatz behält daher die unterschiedlichen Greifbarkeitskarten und führt aufgabenspezifische Fusionsfunktionen ein.

Zuerst soll der Fall bimanueller Manipulation betrachtet werden, also die Fusion des Arbeitsraums von linken und rechten Arm. Zwei Greifbarkeitskarten für einen bimanuellen Manipulator sind in Abb. 6.6 visualisiert, die Auswirkungen der unterschiedlichen Basispositionen des Arms sind klar zu erkennen. Wird die Erreichbarkeit eines Ortes betrachtet, so spielt die Frage eine Rolle, ob es einen Manipulator gibt, der einen Ort erreichen kann. $G_{O,\text{bimanuell}}$ kann über das Maximum definiert werden:

$$G_{O,\text{bimanuell}} = \max(G_{O,\text{links}}, G_{O,\text{rechts}}) \tag{6.4}$$

Muss für die Aufgabe ein Objekt mit beiden Händen gegriffen werden, so muss ein geeigneter Ort für beide Arme erreichbar sein, das Minimum bietet sich zur Fusion an. Die vorgeschlagenen Neudefinition von g_0 wird durch folgende Fusion approximiert:

$$G_{O,\text{bimanuell}} = \min(G_{O,\text{left}}, G_{O,\text{right}}) \tag{6.5}$$

Für beliebige Posen von Objekten ohne spezifisches Hintergrundwissen kann folgende Funktion verwendet werden:

$$G'_{O,\text{bimanuell}} = \max(G_{O,\text{links}}, \epsilon) \times \max(G_{O,\text{rechts}}, \epsilon) \tag{6.6}$$

mit $0 < \epsilon < 1$. Die Verwendung des Produktes ist über die probabilistische Konjunktion motiviert. Im Vergleich zur einfacheren Summenbildung bevorzugt diese Gebiete mit einer hohen Greifbarkeit stark. Eine reine „Und" Verknüpfunge würde dazu führen, dass die Greifbarkeit nur in Gebieten, die von beiden Armen erreichbar sind, mit einem Wert ungleich 0 bewertet würde. Aus diesem Grund wird die Konstante ϵ als untere Schranke für die Faktoren eingeführt. Der

[1] http://i61p109.ira.uka.de/ObjectModelsWebUI/

Wertebereich von $G'_{O,\text{bimanuell}}$ ist damit jedoch $[\epsilon^2,1]$. Dies lässt sich beheben:

$$G_{O,\text{bimanuell}} = \frac{\max(G_{O,\text{links}},\epsilon) \times \max(G_{O,\text{rechts}},\epsilon) - \epsilon^2}{1-\epsilon^2} \tag{6.7}$$

Für die Fusion der Greifbarkeit unterschiedlicher stabiler Lagen von Objekten ist der aufgabenspezifische Ansatz prinzipiell übertragbar. Um eine Lage in eine andere zu überführen, bietet sich die Fusionsfunktion aus Gleichung 6.5 an. Für die in dieser Arbeit betrachteten Problemstellungen liegt jedoch im Allgemeinen kein aufgabenspezifisches Wissen über die Lage von Objekten vor. Daher soll ein generischer Ansatz verfolgt werden. Betrachtet man die Greifbarkeitskarten für ein Objekt in unterschiedlichen Lagen in Abb. 8.5, so ist eine große Ähnlichkeit der Greifbarkeiten zu erkennen. Dies ist jedoch nur bedingt verallgemeinerbar. Ein Müslikarton etwa kann von vielen Greifen nicht gegriffen werden, wenn er auf seiner größten Seite liegt. Die Greifbarkeit ist dann für diese Orientierung an jedem Ort Null. In anderen Orientierungen gibt es mögliche Griffe, wie in Abb. 6.6 dargestellt. Der Ansatz aus Gleichung Gleichung 6.7 ist in der Lage, diese unterschiedlichen Fälle sinnvoll zu fusionieren, er wird daher zur Fusionsfunktion für n stabile Lagen erweitert:

$$G_{O,\text{Lage}} = \frac{-\epsilon^n + \prod_{i=0}^{n} \max(G_{O,i},\epsilon)}{1-\epsilon^n} \tag{6.8}$$

6.3.3.4. Stabile Lagen

Stabile Lagen eines Objektes werden in dieser Arbeit zur Beschreibung möglicher auftretender Orientierungen eines Objektes verwendet. Sie werden mittels des Verfahrens aus (Xue u. a., 2008b) automatisch aufgrund des geometrischen Modells des Objektes erzeugt. Das Verfahren wird in diesem Abschnitt beschrieben.

Für ein Objekt, für das stabilen Lagen bestimmt werden sollen, wird die konvexe Hülle bestimmt. Diese wird durch koplanare Polygone dargestellt. Es wird angenommen, dass das Objekt eine homogene Masseverteilung besitzt. Unter dieser Annahme kann der Schwerpunkt des Objektes bestimmt werden. Für jedes Polygon der konvexen Hülle wird der Schwerpunkt entlang des Normalenvektors auf die Ebene, in der das Polygon liegt, projiziert. Fällt die Projektion in das Polygon, so ist es ein Kandidat für eine stabile Ebene. Anderenfalls wird das Polygon nicht weiter betrachtet.

Für alle Kandidaten wird nun ein Stabilitätsmaß bestimmt. Dazu wird der maximale Kegel, in dem die Auslenkung des Objektes zur Normalen liegen darf, ohne dass das Objekt kippt, bestimmt. Eine Stabilitätsbetrachtung aufgrund des Raumwinkels des Kegels führt zu einem Stabilitätskriterium, das wiederum die Festlegung eines Schwellwertes für die maximale Aus-

Name	Typ	Abdeckung
Erzeugt		
IstFrei(Ort)	Relation	vollständig
Verwendet		
Aktion	Entität	
Ort	Entität	

Tab. 6.3.: Symbole, die durch den „Aktionen-Symbolisierungsoperator" erzeugt und verwendet werden.

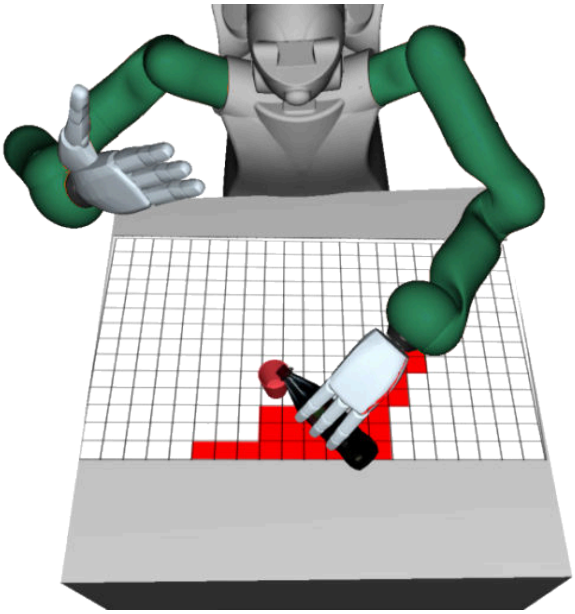


Abb. 6.7.: Freiraumanforderungen für eine Einschenk-Aktion. Nur eine einzelne Armkonfiguration der Manipulation ist dargestellt. Orte, auf die die Projektion des Arms fällt und für die „IstFrei" Symbole erzeugt werden, sind rot markiert.

lenkung erlaubt. Wird dieser Schwellwert unterschritten, so wird der Kandidat im Filterschritt verworfen.

Ein unerwünschter Effekt des Filterschrittes ist, dass alle Lagen auf zylindrischen oder kugelförmigen Seiten des Objektes verworfen werden. Für einen Ball werden keine Lagen erzeugt. Daher wird in einem dritten Schritt eine Anzahl von Ebenen gleichverteilt im Raum der Orientierungen erzeugt. Würden z.B. sechs Lagen gewählt, so entstünden gerade die Ebenen mit den Raumachsen bzw. deren Negative als Normalenvektoren. Lagen, die einem schwächeren Stabilitätskriterium nicht entsprechen, werden verworfen.

6.4. Symbolisierung der Aktionen

Der Symbolisierungsoperator für Aktionen erzeugt die Freiraumbedingungen, die erfüllt sein müssen, damit eine Aktion kollisionsfrei ausgeführt werden kann. Dabei handelt es sich um ge-

nerische Vorbedingungen, die für jede Manipulationsaktion erzeugt werden. Aktionsspezifische Vorbedingungen werden von diesem Symbolisierungoperator nicht erzeugt. Zur Modellierung der Anforderungen von Aktionen erzeugt der Aktions-Symbolisierungsoperator das „Ist Frei"-Symbol als Vorbedingung für die Aktion. Er basiert auf bekannten Aktions- und Ortssymbolen. Eine systematische Darstellung ist in Tabelle 6.3 zu sehen.

Das Verfahren zur Erzeugung der „Ist Frei" Symbole basiert auf kontinuierlichen Modellen, insbesondere den geometrischem Modell und der Armkinematik. Es ist für beliebige Manipulationsplaner verwendbar, die eine Armtrajektorie erzeugen und damit zur Konkretisierung einer Aktion verwendet werden können. In eine ansonsten leere, simulierte Szene werden die für die Aktion relevanten Objekte an ihrer aktuellen Position eingefügt. Der Manipulationsplaner wird auf die Szene angewandt, sein Ergebnis ist eine Arm-Trajektorie. Diese wird zeitlich zu einer Menge von Konfigurationen abgetastet. Der Manipulator wird für das Verfahren mit umschließenden Quadern für jedes starre Segment dargestellt. Diese werden wiederum durch eine Punktwolke, die durch homogene Abtastung ihrer Oberfläche entsteht, angenähert. Mittels Vorwärtskinematik wird jede Armkonfiguration auf die umschließende Punktmenge abgebildet. Ist die Abtastdichte (räumlich und örtlich) hoch genug gewählt, so muss jede Kollision des Armes dazu führen, dass auch einer der umschließenden Punkte in ein Objekt fällt. Daher ist die Aktion kollisionsfrei, wenn die Vereinigungsmenge der Punkte keinen Schnitt mit Objekten in der Szene hat.

Im nächsten Schritt werden die erzeugten Punkte entlang der Z-Achse in das Raster der Tischdiskretisierung (vgl. Abschnitt 6.3) projiziert. Für jedes Feld des Rasters muss lediglich der niedrigste Punkt über dem Tisch berücksichtigt werden. Der Manipulationsplaner erzeugt eine sicher kollisionsfreie Trajektorie, daher können Punkte, die in oder unter den Tisch fallen, als Approximierungsfehler verworfen werden. Für jeden Ort im symbolischen Modell wird nun der Punkt im korrespondierendem Feld des Rasters betrachtet. Ist dieser niedriger als ein Schwellwert, z.B. der Höhe des höchsten verwendeten Objektes, so wird für den Ort eine „Ist Frei" Vorbedingung erzeugt.

Im Allgemeinen sind Manipulationsplaner in der Lage, mehr als eine mögliche Ausführung für eine Aktion zu erzeugen. Daher wird das Verfahren um einen Zyklus erweitert um „Ist-Frei" Vorbedingungen für unterschiedliche mögliche Ausführungen zu erzeugen. Die unterschiedlichen Ausführungen können dann als unterschiedliche Instanzen der Aktionen betrachtet werden, wobei die Ausführung von genau einer Instanz die Vorbedingung für das Symbol der eigentlichen Aktion ist. Damit kann der symbolische Planer die Instanz auswählen, die am wenigsten Veränderung der Szene erfordert.

Der verwendete Manipulationsplaner muss so beeinflussbar sein, dass er mehrere Ausführungen einer Aktion erzeugt. Dies kann bei randomisierten Verfahren ohne weiteres Zutun der Fall

sein, andere Verfahren sind durch Parametrisierung beeinflussbar. Ist auch das nicht möglich, so können zusätzliche Hindernisse in die Szene eingefügt werden, um den Planer zu einer Veränderung zu zwingen. Diese Vorgehen muss für die Planer spezifisch erstellt werden.

Für die Implementierung des Verfahrens bieten sich eine Optimierung an. Wenn der verwendete Planer numerische Vergleiche erlaubt, ist es sinnvoll, kein absolutes „Ist Frei" Symbol zu verwenden, sondern für jeden Ort eine Relation zur maximal erlaubten Objekthöhe zu erzeugen. Die Objekthöhe wird dementsprechend auch mit den Objektsymbolen gespeichert. Ein Objekt muss nur dann entfernt werden, wenn es höher ist als die erlaubte Höhe.

Die gewählt Modellierung ermöglicht es, dem symbolischen Planer eine Umgebung zu erzeugen, in der mindestens eine der im Aktions-Symbolisierungsoperator erzeugten Ausführungen anwendbar ist. Die Konkretisierung auf die Trajektorie, die tatsächlich ausgeführt wird, geschieht wieder durch den Manipulationsplaner. Dieser kann auch eine andere ausführbare Trajektorie erstellen. Es gibt keine Anforderung, dass diese der ursprünglichen ähnlich sein muss.

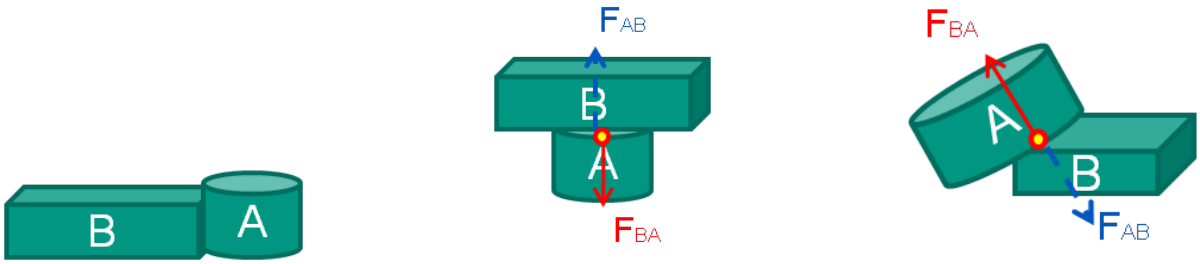
Ein Nachteil des Verfahren ist, dass die Aktionen von den Objektpositionen abhängig sind. Es muss entweder angenommen werden, dass die relevanten Objekte nicht bewegt werden, etwa weil Aktionen aufgrund kinematischer Anforderungen nur in einem eng begrenzten Bereich ausführbar sind. Oder das Verfahren muss in Abhängigkeit vom Ort für die Aktion in mehreren Durchläufen unterschiedliche Vorbedingungen erzeugen.

6.5. Symbolisierung der Szenenmechanik

Die Szenenmechanik wird durch mechanische Relationen beschreiben. Sie modellieren die Auswirkungen der Manipulation in der Szene auf Objekte, die nicht direkt an der ausgeführten Aktion beteiligt sind. In diesem Abschnitt werden geeignete Symbole und deren Bedeutung untersucht. Im Anschluss wird ein Verfahren zur automatischen Erzeugung der Symbole beschrieben.

6.5.1. Auswahl

Auswirkungen von Manipulationsaktionen auf Objekte entstehen durch wirkende Kräfte. Diese können nur zwischen sich berührenden Objekten wirken. Dementsprechend werden durch die Manipulationsplanung Berührungen zwischen Roboter bzw. dem manipulierten Objekt und weiteren Objekten durch Methoden der kollisionsfreien Bahnplanung vermieden. Dies ist nicht möglich, wenn Objekte sich bereits zu Beginn der Manipulation in Kontakt befinden. Im Folgenden werden binäre Relationen untersucht, die Zusammenhänge zwischen zwei Objekten beschreiben. Konstellationen mit mehr Objekten werden durch paarweises Anwenden der Rela-



(a) (b) (c)

Abb. 6.8.: Kräfte zwischen Objekten in einer statischen Szene: (a) zeigt zwei sich berührende Objekte. Zwischen den Objekten wirkt keine Kraft. In (b) trägt Objekt A Objekt B. Es wirkt eine nach oben gerichtete Kraft auf Objekt B. In (c) lehnt das Objekt A an Objekt B. Es wirkt eine Kraft vom Objekt B auf A, die eine seitwärts gerichtete Komponente enthält. Diese hält zusammen mit einer durch den Tisch ausgeübten Kraft den Ort von A.

tionen beschrieben. Werden Objekte durch Aktionen des erzeugten Plans in der Szene platziert, so geschieht dies kollisionsfrei auf dem Tisch. Für die so erzeugte Szene müssen daher keine neuen mechanischen Relationen erzeugt werden; eventuell bestehende Relationen mit Bezug zum platzierten Objekt müssen entfernt werden.

6.5.1.1. Kraftbasierte mechanische Relationen

Es werden Symbole untersucht, die beschreiben, ob sich Objekte in Kontakt befinden und wie Kräfte zwischen ihnen wirken. Einige Konfigurationen von berührenden Objekten sind in Abb. 6.8 dargestellt. Intuitiv können die Fälle unterschieden werden, in denen Objekte andere Objekte tragen, in diesem Fall wirkt eine vertikale Kraft der Schwerkraft entgegen, wie in Abb. 6.8(b). Dem gegenüber steht der Fall einer eher horizontal wirkenden Kraft. In diesem Fall spricht man von aneinander lehnenden Objekten.

Allerdings führt diese Unterscheidung auch zu Uneindeutigkeiten wie in Abb. 6.9(c). Bei so einer Konstellation ist intuitiv nicht klar, ob die Schachtel getragen wird, oder ob sie an der Dose lehnt. Auch bei der Objektkonfiguration in Abb. 6.8(a) ist es möglich, dass durch weitere Objekte Kräfte auf A und B ausgeübt werden, sodass trotz der Lage auf einer horizontalen Ebene noch eine Kraft zwischen den Objekten wirkt. Auch in diesem Fall ist es nicht direkt schlüssig, welcher Relation die Konstellation zugeordnet werden soll.

Auf der anderen Seite bringt eine Unterscheidung aufgrund der Wirkungsrichtung keinen Mehrwert für die Manipulation. In allen gezeigten Beispielen in Abb. 6.8 und Abb. 6.9 ist es ausreichend, eine geeignete Reihenfolge für die Manipulation zu finden, sodass das Objekt, welches durch das andere Objekt an seinem Ort gehalten wird, zuerst entfernt werden kann. Dafür ist alleine die Kraftrichtung ausschlaggebend. Wirkt sie von Objekt B mit einer Komponente in

(a) (b) (c)

Abb. 6.9.: Konfigurationen, die durch die „Stützt" Relation abgedeckt werden: In Abb.(a) befindet sich die Teeschachtel auf der Schinkendose: *Stützt(Dose, Schachtel)*. In Abb. (b) lehnt die Dose an der Schachtel, weiterhin gillt: *Stützt(Schachtel, Dose)*. In Abb. (c) ist die Unterscheidung zwischen tragen und lehnen uneindeutig. In jedem Fall gillt: *Stützt(Dose, Schachtel)*.

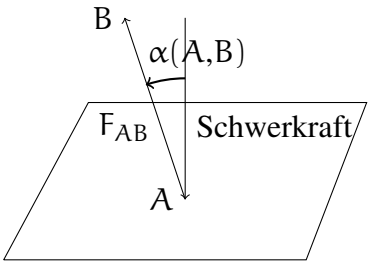


Abb. 6.10.: Richtung der wirkenden Kraft auf ein Objekt mit Winkel α zum Gravitationsvektor.

positiver Z-Richtung auf A, so trägt A B, es gilt Trägt(A,B). Der Winkel α zwischen Kraftrichtung und Normalen (Abb. 6.10) ist damit im Intervall $(-90°, 90°)$. Es genügt, lediglich eine „Stützt" Relation zu erzeugen, die das Wirken einer Kraft und deren Richtung beschreibt:

Definition 6.3

$$\mathit{Stützt}'(A,B) \Longleftrightarrow \exists F_{AB} \neq 0 \wedge \alpha(A,B) \in (-90°, 90°)$$

mit $\alpha(A,B)$ wie in Abb. 6.10 skizziert.

Nachteilhaft an der Definition von von „Stützt" ist allerdings, dass für $\alpha(A,B) = \pm 90°$ keine Relation definiert ist. Für den Fall ist nicht klar, welches Objekt welches stützt. Dies ist für den Fall des Anlehnens in Abb. 6.9(b) relevant. Hier kann anhand der Kraft nicht entschieden werden, welches Objekt welches stützt. Im Sinne einer stabilen Manipulation wird daher angenommen, dass die Objekte gegenseitig für ihre Position notwendig sind, sie stützen sich also gegenseitig. Die Definiton der „Stützt" Relation wird mit einem Schwellwert ϵ angepasst, um dies wiederzugeben:

Definition 6.4

$$\mathit{Stützt}(A,B) \Longleftrightarrow \exists F_{AB} \neq 0 \wedge \alpha(A,B) \in (-90° - \epsilon, 90° + \epsilon) \tag{6.9}$$

Mit dieser Definition erzwingt man eine stabile Szene, verbietet jedoch gleichzeitig auch mögliche Manipulationsaktionen. Um Objekte zu identifizieren, zwischen denen Kräfte wirken, deren Bewegung jedoch keine unerwünschten Folgen hat, wird eine Beschreibung über die wirkenden Kräfte hinaus benötigt.

6.5.1.2. Bewegungsbasierte mechanische Relationen

Im Folgenden werden Relationen untersucht, welche die Effekte der Bewegung eines Objektes auf ein anderes beschreiben. Wird ein Objekt A, das mit einem anderen Objekt B in Kontakt ist, bewegt, so können unterschiedliche Effekte auftreten, die sich in Bewegungen des Objektes B ausdrücken:

1. B bewegt sich nicht. Dies kann der Fall sein, wenn A und B lediglich durch eine seitliche Berührung in Kontakt sind (Abb. 6.8(a)) und A vom Kontaktpunkt entfernt wird.
2. B beschreibt die selbe Bewegung wie A. B steht auf A und wird mit diesem bewegt. Dies kann nur geschehen, wenn die tragende Oberfläche in der Horizontalen bleibt. Oder A wird in Richtung von B bewegt und schiebt B vor sich her.
3. B beschreibt eine kürzere Bewegung als A. Z.Bsp. ebenfalls durch Schieben von B durch A, aber mit Kontaktverlust in der Bewegung.
4. B beschreibt eine größere Bewegung als A. In diesem Fall muss ein zusätzlicher Antrieb für die weitere Bewegung von B vorhanden gewesen sein. Dieser stammt aus potentieller Energie von B, B muss also gefallen sein.

Als weiterer Aspekt kann die Bewegung von B in Z-Richtung betrachtet werden. Bei einer Bewegung, die auf die horizontale Ebene beschränkt ist, besteht keine Gefahr, dass Objekte beschädigt werden. Unter Berücksichtigung der ursprünglichen Motivation, ein symbolisches Modell für einen Planer zu erzeugen, ist dieser Aspekt jedoch nicht relevant. Die wesentliche Anforderung, die eine Aktion erfüllen muss, ist die Kontrolliertheit. Aus Startzustand und der Aktion muss der Ergebniszustand ableitbar sein. Das ist für 3) und 4) nicht der Fall, daher ist es nicht nötig, diese Fälle weiter zu unterscheiden. Fall 1) und 2) dagegen sind für die Manipulation relevant. Aufgrund der Diskussion werdend die folgenden Relationen definiert:

- *Starr(*A,B*)* analog zu Fall 1). Trotz Berührung hat die Bewegung von A keine Auswirkung auf B. A kann manipuliert werden.

(a) (b) (c)

Abb. 6.11.: Bewegungsbasierte mechanische Szenenrelationen. Alle dargestellten Konstellationen enthalten Objekte, die sich in der „Unstabil" Beziehung befinden. In (a) befinden sich die gestapelten Objekte in einem instabilen Gleichgewicht. In (b) lehnen die Objekte symmetrisch aneinander: Unstabil(A,B), Unstabil(B,A) ebenso wie Stützt(A,B), Stützt(B,A), da die Kraft zwischen den Objekten in horizontaler Richtung wirkt. In der Szene in (c) haben die beiden zylindrischen Objekte keinen direkten Kontakt, stehen aber über das gemeinsam tragende Objekt trotzdem in einer „Unstabil" Relation.

- *Stabil(*A,B*)* analog zu Fall 2). B beschreibt die selbe Bewegung wie A, B wird von A stabil getragen. A kann manipuliert werden, die Aktion muss jedoch die Effekte der Bewegung auf B berücksichtigen.
- *Unstabil*(A,B) Wird A bewegt, so bewegt sich B, aber nicht wie 1) oder 2). Über das genaue Ergebnis kann keine Aussage gemacht werden, ein kontrollierte Manipulation ist nicht möglich.

Mit den beschriebenen Relationen kann die Situation aus Abb. 6.9(a) aufgelöst werden. Weitere Beispiele für die „Unstabil" Relation sind in Abb. 6.11 gezeigt. In Abb. 6.11(a) stellt die durch Unstabil/Stabil dargestellte Beziehung den selben Sachverhalt dar, der auch durch die „Stützt" Relation dargestellt wird. In Abb. 6.11(b) ist die Situation symmetrisch, beide Objekte haben eine „Unstabil" Relation zueinander. Um diese Situation aufzulösen, muss der Planer eine bimanuelle Aktion verwenden, die beide Objekte gleichzeitig greift. Dies kann durch entsprechende Vorbedingungen modelliert werden. Abb. 6.11(c) zeigt die Grenzen der Modellierung auf: Wird hier eines der Zylinderobjekte entfernt, so kippt der Stapel. Dies wird durch die „Unstabil" Relation auch korrekt dargestellt – trotz der fehlenden Berührung; hier gehen die bewegungsbasierten Relationen über die kraftbasierten hinaus. Die Lösung, beide Zylinder gleichzeitig zu entfernen kann das System jedoch nicht finden, da es den veränderten Effekt der simultanen Aktionsausführung auf die tragende Schachtel nicht schließen kann. Die Möglichkeit, die tragende Schachtel mit den Zylindern zu entfernen, kann modelliert werden, sodass auch in dieser Situation eine Lösung existiert.

Die vorgestellten Relationen werden durch den Symbolisierungsoperator der Szenenmechnik erzeugt. Einen Überblick über die Relationen gibt Tabelle 6.4

Name	Typ	Abdeckung
Erzeugt		
stützt(Objekt, Objekt)	Relation	vollständig
starr(Objekt, Objekt)	Relation	vollständig
stabil(Objekt, Objekt)	Relation	vollständig
unstabil(Objekt, Objekt)	Relation	vollständig
Verwendet		
Objekte	Entität	

Tab. 6.4.: Symbole, die durch den „Szenenmechanik-Symbolisierungsoperator" erzeugt und von ihm verwendet werden.

6.5.2. Erzeugung

Die automatische Erzeugung der Symbole der Szenenmechanik basiert auf einer Physiksimulation. Im Folgenden wird auf ihre Eigenschaften und benötigte Modelle eingegangen. Die Erzeugung der mechanischen Relationen geschieht für kraft- und bewegungsbasierte Relationen getrennt, sie werden im Anschluss beschrieben.

6.5.2.1. Physiksimulation und Modelle

Die vorgestellten Relationen zur Beschreibung der Szenenmechanik basieren auf Kräften und daraus resultierenden Bewegungen. Die Simulation solcher Zusammenhänge wird durch Physiksimulationen geleistet. Diese werden oft in Computerspielen verwendet, um die Effekte der Aktionen des Spielers zu berechnen. Dort kommt es nicht auf hohe Genauigkeit an, sondern hohe Performance und weiche Echtzeitfähigkeit. In der untersuchten Anwendung in dieser Arbeit sind die Anforderungen ähnlich. Schnelle Berechnungen werden benötigt, um mehrere Simulationsvorgänge in kurzer Zeit vor der Planung durchführen zu können. Auf der anderen Seite sind die Anforderungen an die Genauigkeit gering, da das Ergebnis lediglich qualitativ betrachtet wird (welche Objekte bewegen sich, wirken Kräfte eher horizontal oder vertikal) und keine quantitative Auswertung (wo kommt ein Objekt zu liegen) stattfindet. Auch, dass lediglich kurze Zeiträume betrachtet werden, erlaubt es, eine grob annähernde Simulation zu verwenden.

In dieser Arbeit wird die Open Source Physik Simulation „Bullet" (Bullet, 2014) und insbesondere deren Starrkörpersimulation eingesetzt. Zur Simulation werden weitere Objekteigenschaften benötigt, die über die zur Manipulationsplanung verwendeten Eigenschaften hinausgehen. So weit wie möglich werden geometrische Eigenschaften verwendet, um sie zu schätzen:

Geometrische Modelle Geometrische Modelle der Objekte werden in der Form von Dreiecksnetzen als vorhanden angenommen. Die Physiksimulation benötigt sie, um die Kon-

taktpunkte zwischen Objekten zu bestimmen, über die Kräfte zwischen den Objekten wirken.

Ort und Geschwindigkeiten Der Ort der Objekte ist aus dem geometrischen Modell bekannt. Da eine statische Szene modelliert wird, sind alle initialen Geschwindigkeiten Null.

Masse, Schwerpunkt und Trägheitsmoment Für jedes Objekt muss seine Masse m bekannt sein. Objektmodelle stammen in dieser Arbeit aus einer externen Datenbank. Es ist möglich, diese um die benötigten objektspezifischen Eigenschaften zu ergänzen. Ist die Masse nicht aus dem Objektmodell bekannt, so kann sie mit der Sensorik des Roboters auch mit dem Verfahren von (Xue u. a., 2008a) bestimmt werden. Dabei greift der Roboter ein Objekt und bestimmt aus den auftretenden Momenten in den Fingern analytisch die Masse des Objektes.

Schwerpunkt und Trägheitsmoment werden durch die Annahme einer homogenen Dichte des Objektes angenähert. Dazu wird das dreidimensionale Volumen des Objektes mit einer Menge P Punkte abgetastet, sodass die Punkte in P im Objekt liegen und darin gleichmäßig verteilt sind. Der Schwerpunkt des Objektes wird darauf wie folgt angenähert:

$$\bar{x} = \frac{1}{|P|} \sum_{i=0}^{|P|} x_i \tag{6.10}$$

Analog gilt für das Trägheitsmoment:

$$\bar{J} = \frac{m}{|P|} \sum_{i=0}^{|P|} |x_i - \bar{x}|^2 \tag{6.11}$$

Diese Schätzungen unterliegen zwar einem nicht zu vernachlässigendem Fehler, da es hier jedoch um qualitative Aussagen zu Eigenschaften der Szene geht, kann dieser in Kauf genommen werden.

Elastizität Die Elastizität $k \in [0,1]$ bestimmt, welcher Anteil der übertragenen Energie eines Stoßes in Form eines Impules zwischen den beteiligten Objekten übertragen wird. Die restliche Energie geht in Form von plastischer Verformung, Reibung im Objekt und damit letztlich in Wärme verloren. Wieder wird in der gegebenen Anwendung kein genauer Wert benötigt. Es wird eine empirisch ermittelte Konstante für alle Objekte verwendet.

Reibung Der Reibungskoeffizient μ bestimmt für zwei Oberflächen A und B das Verhältnis der benötigten Kraft F_R, um die Flächen gegeneinander zu verschieben zur Kraft F_N, die entlang der Normalen der Flächen wirkt: $F_R = \mu_{AB} \cdot F_N$. Die Reibung hängt dabei von den Materialien beider Flächen ab und wird für Materialpaare experimentell bestimmt (Tipler

Materialklasse χ	Reibungskoeffizient μ_χ
Holz	0,5
Papier	0,4
Metall	0,35
Plastik	0,2

Tab. 6.5.: Zur Simulation verwendete Reibungskoeffizienten.

u. a., 2009, Kap. 5, S. 99-100). Die eingesetzte Simulation verwendet ein vereinfachtes Modell. Für jedes Material X wird ein Reibungskoeffizient μ_X festgelegt. Für ein Materialpaar bestimmt das Produkt den gemeinsamen Reibungskoeffizienten $\mu_{AB} = \mu_A \cdot \mu_B$. Eine Unterscheidung zwischen Haft- und Gleitreibung findet nicht statt.

In dieser Arbeit werden die Reibungskoeffizienten für Materialklassen angegeben. Die verwendeten Werte sind in Tabelle 6.5 zu finden.

Gravitation Die Gravitation wird konstant mit 9,81 $\frac{m}{s^2}$ modelliert.

Objektpositionen im Szenenmodell sind abhängig von der verwendeten Sensorik mit Wahrnehmungsfehlern behaftet. Diese führen dazu, dass sich trotz statischer Szene Objekte in der Simulation bewegen können. Gestapelte Objekte haben möglicherweise noch keinen Kontakt, Kräfte zwischen ihnen können nicht beobachtet werden. Um diesem Effekt entgegen zu wirken, wird für jede Szene in der Physiksimulation eine Abkühlungsphase ausgeführt, in der die Simulation die Objekte in eine statische Lage absinken lässt. Weitere Analysen finden auf der so erzeugten statischen Szene statt.

Um den Rechenaufwand bei der Simulation zu reduzieren, erfolgt nun eine Segmentierung der Szene aufgrund der Kräfte zwischen den Objekten. Dazu wird angenommen, das tragende Objekte wie ein Tisch zu groß sind, um vom Roboter bewegt werden zu können. Daher können sie auch keine indirekte „Unstabil" Relation zwischen anderen Objekten erzeugen. Unter dieser Annahme kann es Cliquen von mechanisch voneinander unabhängigen Objekten geben. Diese können unabhängig voneinander simuliert und ausgewertet werden.

Ausgehend von einem beliebig gewählten Startobjekt werden dieses und alle Objekte, auf die es eine Kraft auswirkt, zu einem Segment hinzugefügt. Dies wird rekursive für alle Objekte in dem Segment wiederholt. Das Verfahren ist in Algorithmus 6.2 detailliert beschrieben. Das Ergebnis sind Cliquen von durch Kräfte zusammenhängenden Objekten.

6.5.2.2. Kraftbasierte mechanische Relationen

Die Physiksimulation berechnet auf Basis der beschriebenen Szene die Kräfte zwischen den Objekten. Zur Bestimmung der kraftbasierten „Stützt" Relation bleibt lediglich, die Kräfte zwi-

Algorithmus 6.2 Gieriger Algorithmus zur Erzeugung von mechanisch unabhängigen Objekt-Cliquen

```
1:  function SEGMENTIERE SZENE(Objektmenge)
2:      Entferne unbewegliche Objekte aus Objektmenge
3:      S ← ∅
4:      while Objektmenge ≠ ∅ do
5:          Wähle o ∈ Objektmenge
6:          Objektmenge ← Objektmenge \o
7:          s′ ← ∅
8:          s ← {o}
9:          while s ≠ s′ do
10:             s′ ← s
11:             for x ∈Objektmenge  do
12:                 if ∃y ∈ s : ∧F(x,y) ≠ 0 then
13:                     s ← s ∪{x}
14:                 end if
15:                 Objektmenge ← Objektmenge \{x}
16:             end for
17:         end while
18:         S ← S ∪{s}
19:     end while
20:     return S
21: end function
```

schen den Objektpaaren zu analysieren. Für zwei Objekte A und B können drei Fälle auftreten, die gemäß der Definition 6.4 direkt einer Relation zugeordnet werden können:

1. $|F_{AB}| < \epsilon \Rightarrow \emptyset$

   Keine Kraft, demnach wird keine statische Relation erzeugt.

2. $-90° - \epsilon < \alpha(A,B) < 90° + \epsilon \Rightarrow$ stützt(A,B)

   Objekt A übt eine nach oben gerichtete Kraft auf Objekt B aus, es gilt also stützt(A,B).

3. sonst $\Rightarrow \emptyset$

   Im Falle einer nach unten gerichteten Kraft wird die Relation durch das Paar B,A erzeugt.

Die Vereinigungsmenge aller erzeugter Relationen für alle Objektpaare definiert die vollständige Menge der kraftbasierten Relationen.

6.5.2.3. Bewegungsbasierte mechanische Relationen

Die kraftbasierten mechanischen Relationen können aus einem Zeitpunkt, also einem einzelnen Simulationsschritt, gewonnen werden. Im Gegensatz dazu muss zur Beobachtung von Bewegungen ein Zeitraum simuliert werden. Wie in Abschnitt 6.5.2.1 ausgeführt, ist die Simulation

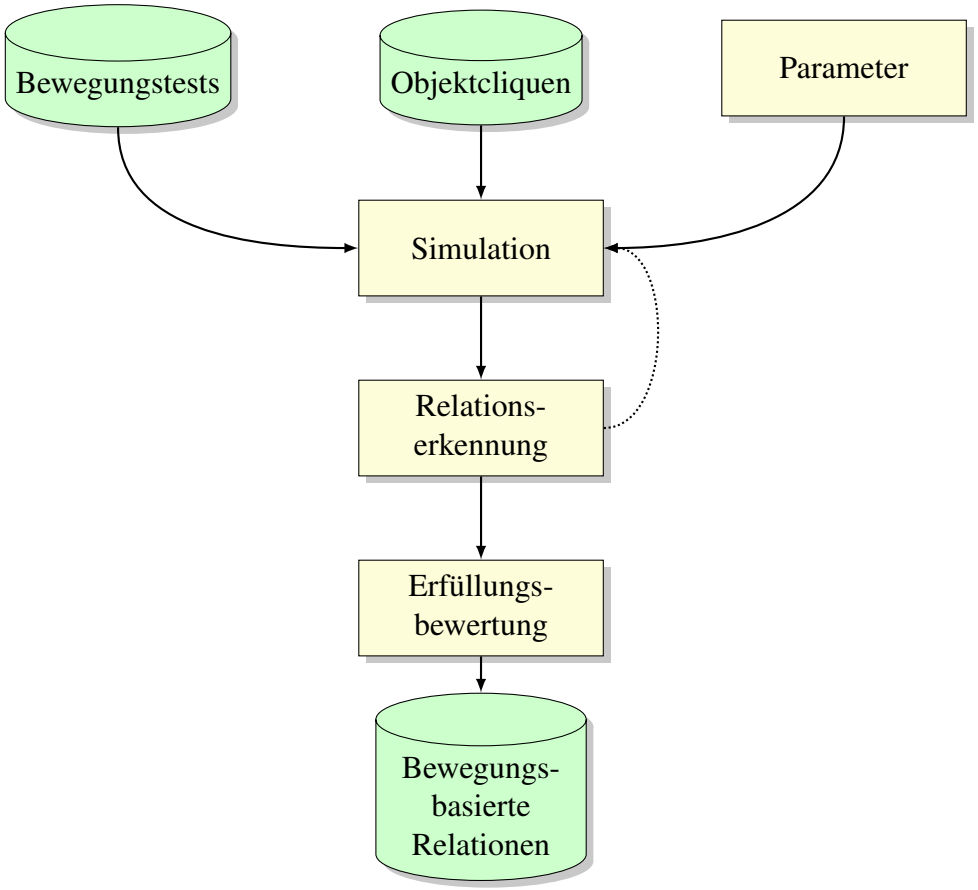


Abb. 6.12.: Prozess zur Erzeugung der bewegungsbasierten mechanischen Relationen. Basierend auf dem Modell der betrachteten Objektclique der Szene werden *Bewegungstests* mit randomisierten Parametern angewandt. Auf der entstehenden Konfiguration der simulierten Szene wird die *Relationserkennung* durchgeführt. Die *Erfüllungsbewertung* fusioniert die Ergebnisse der Relationserkennung für unterschiedliche Simulationsdurchgänge zu den *bewegungsbasierten mechanischen Relationen.*

lediglich eine grobe Annäherung an die Wirklichkeit. Um die Zuverlässigkeit der Erzeugung der bewegungsbasierten mechanischen Relationen zu erhöhen, wird daher nicht eine einzelne Simulation verwendet, sondern das Ergebnis mehrerer Simulationsvorgänge durch den Prozess aus Abb. 6.12 fusioniert.

Die bewegungsbasierten mechanischen Relationen sind nicht nur vom Modell abhängig, sondern auch von den ausgeführten Bewegungen. Ein Objekt auf einem Tablett wird fallen, wenn das Tablett um die Rollachse gedreht wird. Auf der anderen Seite kann auch eine recht instabile Konfiguration eine langsame Bewegung in Z-Richtung überstehen. Daher werden unterschiedliche Bewegungstests verwendet.

Da man an einer Änderung des Ortes eines Objektes interessiert ist, hat jeder Bewegungstest eine translatorische Komponente. Das Beispiel des Tabletts zeigt dagegen, dass durch eine rotatorische Komponente kein nützlicher Erkenntnisgewinn erzielt wird. Ein Bewegungstest wird definiert als ein Tupel

$$(\mathrm{d}, \mathbf{x}, v), |\mathrm{x}| = 1.$$

Dabei ist $\mathbf{x}$ ein dreidimensionaler Vektor der Länge 1, der die Richtung der ausgeführten Bewegung beschreibt. d modelliert die zu bewegende Distanz, v die zu erreichende Geschwindigkeit. Das Objekt wird innerhalb der Stecke auf die Zielgeschwindigkeit beschleunigt und danach wieder zum Stillstand verzögert. Die simulierte Zeitspanne t für einen Test berechnet sich mit

$$\mathrm{t} = \frac{2\mathrm{d}}{v}$$

Entsprechend ergibt sich eine Beschleunigung

$$\mathrm{a} = \frac{2v}{\mathrm{t}}$$

Für $\mathbf{x}$ werden die Achsen des Koordinatensystems verwendet: $(1,0,0),(-1,0,0),(0,1,0),(0,-1,0),(0,0,0),(0,0,1)$. Die negative Z-Achse bietet sich nicht an, da dadurch immer eine Kraft gegen den als unbeweglich modellierten Tisch erzeugt werden würde.

Für die verbleibenden Parameter d und v werden randomisiert Werte erzeugt. Dabei wird ein geringes d verwendet, um Interaktion mit eigentlich unbeteiligten Objekten zu vermeiden. Empirisch wurde ein Intervall von 2 cm bis 5 cm ermittelt. Ebenso wird v aus dem Bereich 0.05 m/s bis 0.25 m/s erzeugt.

Damit sind alle Parameter für die Simulation bestimmt, die Physiksimulation simuliert nun die Bewegung und eine darauf folgende Zeitspanne. Das Ergebnis sind die Posen der beteiligten Objekte, auf denen die Relationserkennung arbeitet. Bewegt sich ein Objekt nicht bzw. weniger als einen Schwellwert, so wird die „Starr" Relation erzeugt, d.h. das Objekt wird nicht weiter

betrachtet. Ist für die verbleibenden Objekte die Differenz der Translation des Objektes und der Translation der simulierten Bewegung größer als ein Schwellwert, so wird eine „Unstabil" Relation erzeugt. Wird der Schwellwert unterschritten, so handelt es sich um eine „Stabil" Relation.

Die Erfüllungsbewertung erzeugt aus den Ergebnissen der unterschiedlichen Durchgänge für ein Objekt die bewegungsbasierten Relationen, die in das Szenenmodell eingetragen werden. Dazu wird für die Objektpaare, für die Relationen bestehen, die beschreibende Relation durch einen Mehrheitsentscheid ermittelt. Kommt es zu einem Patt, wird aus den am Patt beteiligten Relationen die gewählt, die an folgender Prioritätenliste zuerst aufgeführt ist: „Unstabil", „Stabil", „Starr".

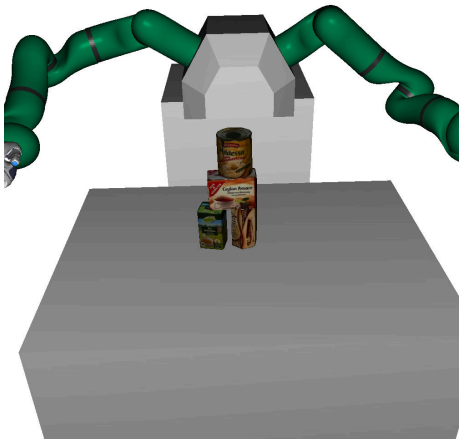
6.5.3. Szenengraph

Die Summe der erzeugten mechanischen Relationen ergibt einen Graphen, der die Mechanik der Szene beschreibt. Ein Beispiel für eine Szene ist in Abb. 6.13 dargestellt. Es ist zu erkennen, dass die gesamte Szene vom Tisch abhängt, auf dem sie aufgebaut ist. Am oberen Ende des Graphen befinden sich die Objekte „CeylonTee" und „InstantSuppe", die keine weiteren Objekte stützen und daher manipuliert werden können. Auch die Unterteilung in zwei Objektcliquen ist im Graphen modelliert. Der Szenengraph dient vor allem als Werkzeug zur Visualisierung der Ergebnisse des Symbolisierungsoperators der Szenenmechanik. Weitere Szenen und ihre Graphen sind in Kap. 8 zu finden.

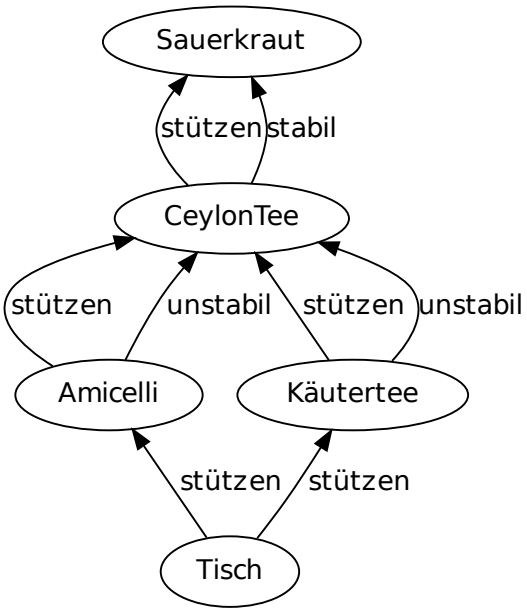
6.6. Fazit

In diesem Kapitel wurde ein Prozess zur automatischen Erzeugung einer symbolischen Beschreibung einer Szene für die Planung von Manipulationssequenzen entwickelt. Das Verfahren ist in der Lage, variable Szenenkonfigurationen abzubilden. Um die unterschiedlichen Symboltypen, die in Kap. 5 identifiziert wurden, berücksichtigen zu können, wurden Symbolisierungsoperatoren eingeführt. Es wurden Methoden zur Umsetzung der Symbolisierungsoperatoren für die Objekte, den Arbeitsraum, Aktionen und die Szenenmechanik vorgestellt. Die Szenenmechanik ist dabei besonders hervorzuheben, da ihre Verwendung im Modell es ermöglicht, Effekte auf Objekte zu berücksichtigen, obwohl das Objekt und der Effekt nicht direkt durch das Aktionsmodell erfasst werden.

(a)



(b)



(c)

Abb. 6.13.: Beispiel für einen erzeugten Szenengraph. Die Szene ist in (a) gezeigt. Der Szenengraph ist ist in (c) dargestellt. Für den Tisch werden keine bewegungsbasierten Relationen erzeugt. Die „starr” Relation ist der Übersichtlichkeit halber vernachlässigt.

7. Konkretisierung von Manipulationsaktionen

Der erstellte symbolische Plan wird im Schritt der Konkretisierung auf kontinuierliche Parameter und Trajektorien abgebildet. Diese müssen an geeignete Ausführungskomponenten weiter gegeben werden, die wiederum die Hardware des Roboters parametrieren. Das Vorgehen dazu wird in Abschnitt 7.1 beschrieben.

Während der Ausführung wird diese anhand von gelernten Bewertungen in Echtzeit überwacht, um auf Abweichungen reagieren zu können. Das dazu entworfene Lernverfahren wird in Abschnitt 7.2 vorgestellt.

7.1. Konkretisierung von Aktionen

Die Konkretisierung muss drei Teilaufgaben lösen:

1. Ersetzen diskreter Parameter durch kontinuierliche, aktuelle Werte aus dem kontinuierlichen Modell.
2. Übertragen diskreter Aktionen in ein kontinuierliches Ausführungssystem. Dabei dürfen Abhängigkeiten der Aktionen untereinander nicht verletzt werden.
3. Zuordnen von kontinuierlichen Trajektorien zu den Aktionen.

Weiter ist der Vorgang der Konkretisierung mit der Sequenzierung der geplanten Aktionen verknüpft. Zur Ausführung von Handlungssequenzen auf Servicerobotern wurden in (Knoop, 2007) die Flexiblen Programme vorgestellt. Diese wurden in Abschnitt 2.1.3 beschrieben und lösen die Sequenzierung unter Einhaltung der Abhängigkeiten der Aktionen. Daher wird zur Konkretisierung und Ausführung ein auf den Flexiblen Programmen basierendes System entwickelt. Es ist Abb. 7.1 skizziert. Weitere Details zu dem System, insbesondere zu den hier nicht vorgestellten unteren Ebenen finden sich in (Rühl u. a., 2009).

Aus dem symbolischen Planer werden erstellte Aktionen vorlagenbasiert auf Flexible Programme abgebildet. Die geplanten Strukturen enthalten Sequenzierung und Parallelität zwischen Aktionen, beides kann in FPs dargestellt werden. Des Weiteren enthält jedes Flexible Programm einen Satz an Parametern, denen Werte zugewiesen werden. Diese Parameter werden aus den Variablen der vom Planer erzeugten Tokens generiert. Sie können insbesondere aus

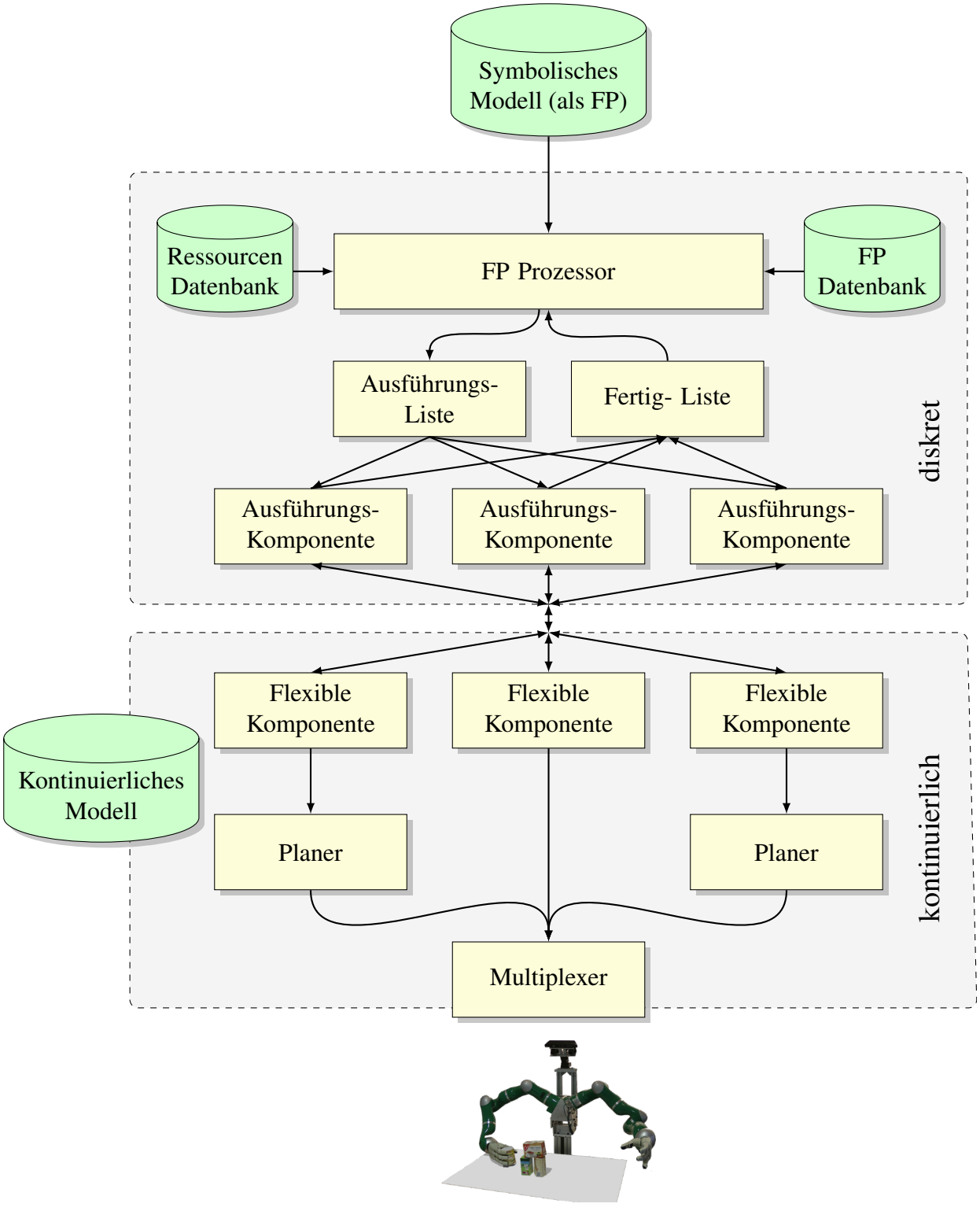


Abb. 7.1.: Überblick über die Struktur des Systems zur Konkretisierung und Ausführung.

den semantischen Anhängen erzeugt werden. Parameter mit einer besonderen Bedeutung sind die Art der Aktion und die ausführende Ressource. Das Wissen über die Zuweisung einer geeigneten Aktion kann in der Planungsdomäne spezifiziert werden. Die zu nutzenden Variablen sind durch ein Prefix gekennzeichnet. Die Konkretisierung ist damit für Planung und Ausführung transparent. Sie muss insbesondere nicht an veränderte Domänen angepasst werden, da das nötige Wissen bereits in der Planungsdomäne kodiert ist.

Der FP Prozessor in Abb. 7.1 erzeugt aus dem Flexiblen Programmen eine Reihe von Ereignissen, die durch das Zusammenspiel von Ausführungskomponente und flexibler Komponente in das zyklusbasierte System der Robotersteuerung im zeitlich Kontinuierlichen abgebildet wird. Die detaillierte Funktionsweise ist hier nicht von Interesse, sie ist in (Rühl u. a., 2009) beschrieben.

In den Flexiblen Komponenten ist darüber hinaus das Wissen vorhanden, wie das aktuelle Weltmodell in die Ausführung eingebracht wird. So kann es eine Aktion, die sich auf ein durch eine Id identifiziertes Objekt bezieht, mit der aktuellen Objektpose parametrieren. Weiter erfolgt hier die Zuordnung von geplanten Parametern zu den Parametern der ausführenden Komponente.

Unterhalb der Flexiblen Komponenten befinden sich die unterschiedlichen Manipulationsplaner. Sie erzeugen aufgrund ihrer Parametrierung und des aktuellen kontinuierlichen Modells Trajektorien. Die Trajektorien werden von ihnen direkt weiter an den Roboter zur Ausführung gegeben. Auf diese Weise ist sichergestellt, dass die Manipulationsplanung im Kontinuierlichen immer auf dem aktuellen Modell durchgeführt wird.

7.2. Ausführungsüberwachung von Manipulationsaktionen

In Abschnitt 4.3.3 wurden die praktischen Grenzen des Konzeptes im Zusammenspiel zwischen symbolischen Plan und dessen Ausführung diskutiert. Diese führen zuerst zu geringen Unterschieden in der geplanten Handlungssequenz und deren Ausführung. Sie können im einzelnen Schritt des Planes vernachlässigt werden. Durch das Aneinanderreihen von geplanten Aktionen werden sie jedoch nichtlinear wachsen. Ebenfalls können einzelne Aktionen fehlschlagen, also nicht den gewünschten oder weitere unerwünschte Effekt liefern.

Als Gegenmaßnahme wurde im vorigen Abschnitt der Ansatz vorgestellt, den Plan während der Ausführung um aktuelles Wissen aus dem kontinuierlichen Modell zu ergänzen. Dies kann jedoch nur einzelne Werte adaptieren. Die Ausführung muss zielgerichtet bleiben, auch wenn die aktuelle Szene nicht mehr der vom Planer erwarteten entspricht. Daher ist es darüber hinaus bei großen Abweichungen notwendig, die geplante Handlungssequenz an die Szene anzupassen. Die Methode dazu liefert das in dieser Arbeit vorgestellte Planungskonzept. Der Ansatz, Planer bei Abweichungen neu anzustoßen ist in der Literatur bekannt und wird als *Replanning*

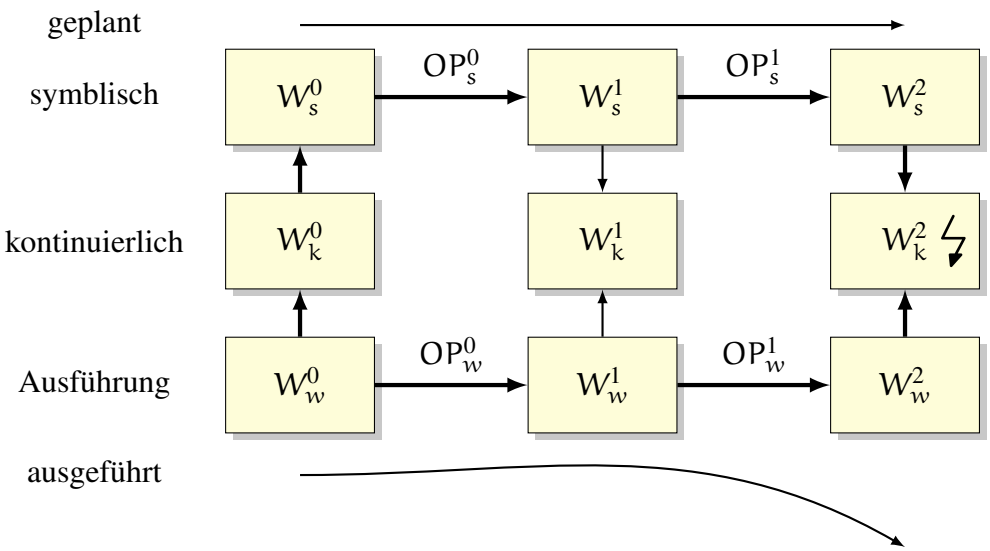


Abb. 7.2.: Abdriften von Plan und Ausführung: Der Zustand W_w^0 wird durch Perzeption und Symbolisierung auf den Weltzustand W_s^0 abgebildet. Daraus wird der Plan OP_s^0, OP_s^1 erzeugt, der im Symbolischen den Zustand W_w^2 erreicht. Der geplante Verlauf einer Eigenschaft der Szene ist am oberen Bildrand visualisiert. Der Verlauf der Eigenschaft während der Ausführung des Plans OP_w^0, OP_w^1 ist am unteren Bildrand visualisiert. Er weicht am Ende des Plans vom geplanten Zustand ab. Wird die Welt zu diesem Zeitpunkt durch die Perzeption in das konkrete Modell abgebildet und zum anderen der geplante Weltzustand W_s^2 konkretisiert, so ergibt sich ein Widerspruch. Die Erkennung solcher, durch das vorgestellte Metamodell nicht abgedeckten Widersprüche ist Aufgabe der Ausführungsüberwachung.

bezeichnet. Um dieses nutzen zu können, ist die Frage zu beantworten, wann eine Neuplanung stattfinden muss. Zwischen den Aktionen wird dies durch Überprüfen der Vor- Nachbedingungen erreicht. Darüber hinaus wird im Folgenden ein Verfahren vorgestellt, dass auch die Einhaltung nicht symbolisch beschriebener Währendbedinungen erkennt und damit auch auf von generischen Manipulationsplanern erzeugte Aktionen skalieren kann.

7.2.1. Analyse

Ein Effekt von fehlschlagenden Manipulationsaktionen sind Kollisionen. Roboterhardware wie der KUKU Leichtbauarm oder die SCHUNK Antrophomorphe Hand (Liu u. a., 2008) besitzen Momentensensoren, die wirkende Momente und Kräfte in Echtzeit bestimmen können. Ebenso können mit dem dynamischen Modell eines Roboters Erwartungswerte für Kräfte und Momente erzeugt werden. Aus beiden zusammen lassen sich die auf den Roboter einwirkenden externen Kräfte schätzen. Diese können verwendete werden, um Kollisionen zu erkennen. Der Ansatz der Ausführungsüberwachung durch Beobachtung der wirkenden Kräfte wird in diesem Abschnitt zu einem Verfahren erweitert, das die Ausführung beliebiger, auch kollisionsbehafteter Manipulationsaktionen überwachen kann.

Manipulationsaktionen sind im Allgemeinen nicht kollisionsfrei; sie haben jedoch spezifische Einschränkungen, die zulässige Kräfte definieren. Wird z.B. eine Tür geschlossen, so werden im Moment der Berührung zwischen Schnapper und Schloss kurzfristig deutlich größere Kräfte auftreten, als während der Bewegung der Tür. Externe Kräfte hängen von der Art der Aktion und ihrem Zustand ab.

Unabhängig davon, ob die überwachte Aktion kollisionsfrei geplant ist oder nicht, können durch große auftretende Kräfte Schäden am Roboter und an Objekten entstehen. Es ist daher notwendig, Fehler in Echtzeit zu erkennen. Im Folgenden werden dazu Sensormessungen zu einzelnen Zeitpunkten betrachtet. Die Analyse einer Messreihe einer vollständigen Ausführung ist unter der diskutierten Bedingung nicht sinnvoll.

Kräfte zwischen Roboter und manipulierten Objekten führen zu Momenten in den Gelenken des Manipulators und Greifers. Bei Verwendung einer Impedanzgegelung (s. Abschnitt 2.7) führen diese weiter zu einer Differenz zwischen kommandierter und erreichter Konfiguration. Die interne Sensorik des Roboters erzeugt während der Ausführung einer Aktion eine Sequenz von Datenpunkten:

$$S = (d_0, ..., d_m)$$

$$d_i = (t_i, v_{i,0},...,v_{i,n}); i \in 0...m$$

Dabei is n die Dimensionalität des Datenraumes, m die Anzahl an Punkten in der Sequenz und t_i ein Index, der den Verlauf der Zeit widerspiegelt. Eine Sequenz $v_{i,j}$ mit konstanten j enthält die Messungen eines spezifischen Sensorwertes und wird als *Kanal* bezeichnet:

Definition 7.1 ***Kanal***

$$K = (v_{0,j}, v_{1,j}, ..., v_{m,j})$$

In einem Zeitfenster von t_k bis t_l kann man folgende Eigenschaft definieren:

Definition 7.2 ***Zeitinvarianz*** *Ein Kanal K ist zeitinvariant von* k *bis* l*, falls gilt:*

$$\exists \tau : \forall i \in k...l : |\tau - v_{i,c}| < \epsilon$$

wobei die Schwelle ϵ einen Rauschanteil im Signal erlaubt. Kanäle, die nicht zeitinvariant sind, heißen entsprechend zeitvariant.

Ist man in der Lage, einen zeitinvarianten Kanal zu identifizieren, der im Falle eines Fehlschlagens der Aktion seine Invarianz verliert, so ist die Ausführungsüberwachung auf die Prüfung des Wertebereichs reduziert. Ein solcher Kanal wird *relevant* genannt. Betrachtet man den Graphen der Momente und Gelenkwinkel in den Fingern eines Greifers in Abb. 7.3, so ist zu erkennen, dass es für die vollständige Aufgabe keinen visualisierten Kanal mit dieser Eigenschaft gibt. Bis

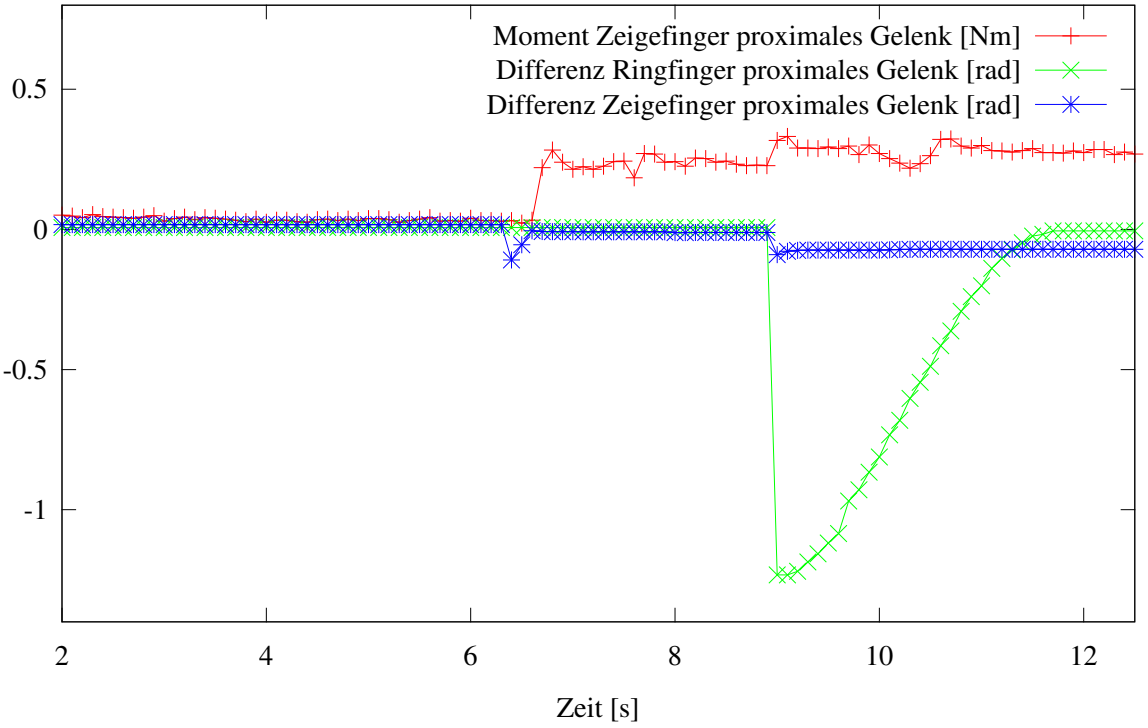


Abb. 7.3.: Diagramm von auftretenden Momenten und Gelenkwinkeldifferenzen während eines Greifvorgans.

zu Sekunde 6 verläuft die Aktion kollisionsfrei; zu diesem Zeitpunkt berührt der Finger das Objekt. Unterteilt man die Aktion dort in zwei Teilaktionen, so erfüllt das Moment im Zeigefinger (rot dargestellt) das Kriterium der Relevanz; es zeigt an, ob ein Finger das Objekt berührt.

Basierend auf diesen Überlegungen müssen für die Ausführungsüberwachung zwei Teilprobleme gelöst werden. Zum einen muss eine Segmentierung geeignete Abschnitte der Aktionen zusammenfassen. Zum anderen müssen in den einzelnen Segmenten relevante Kanäle und zulässige Abweichungen erkannt werden.

7.2.2. Segmentierung

Die Segmentierung der Aktionen wird durch den symbolischen Planer vorgenommen. Die Ebene der atomaren Aktionen muss dafür durch einen Experten so gewählt werden, dass sinnvolle, semantisch zusammenhängende Teilaktionen erzeugt werden. Dies ermöglicht ebenfalls, die Parametrierung der Ausführungsüberwachung durch vom Planer erstellte Hilfsaktionen setzen zu lassen.

Beispielsweise ermöglicht die gewählte Aufteilung der Transportaktion die Unterscheidung der Segmente „Anrücken", „Pick", „Move", „Place" und „Abrücken". Dabei kann überwacht werden, dass „Anrücken" und „Abrücken" kollisionsfrei ausgeführt werden. Bei„Move" dagegen

müssen Kräfte auftreten, die auf das Objekt zurückzuführen sind, ansonsten treten keine Kräfte zwischen Roboter und Umwelt auf. „Pick" und „Place" sind die Phasen des Übergangs, in welchen keine konstante Aussage über Kräfte zwischen Roboter und Objekt erfolgen kann, da sich hier zwingend der den Erfolg bestimmende Kanal ändert.

Für neue Aktivitäten muss eine geeignete Segmentierung erstellt werden. Dies erfolgt im vorgestellten System durch einen Experten. Möglich wäre es darüber hinaus, ein Verfahren zur automatischen Segmentierung zu verwenden. In Pardowitz (2007) wird ein solches vorgestellt. Es zerlegt Vorführungen eines Benutzers und erzeugt eine symbolische Beschreibung der Segmente.

7.2.3. Verfahren zur Klassifikation

In diesem Abschnitt wird die Klassifikation von Datenpunkten innerhalb eines Segmentes behandelt. Dazu werden Messwerte, die Kandidaten für relevante Kanäle sind, identifiziert. Sie bilden den Merksmalsraum, auf den ein überwachtes Lernverfahren angewandt wird.

7.2.3.1. Merkmalsraum

Die zur Überwachung zu verwendenden Merkmale müssen die Interaktion zwischen Manipulator und Umwelt widerspiegeln. Aus diesem Grund werden die Messungen von Kräften und Momenten betrachtet. An dieser Stelle wird weiter auf die Impedanzregelung des Roboters zurückgegriffen. Wie in Abschnitt 2.7 beschrieben, koppelt diese Positionsdifferenzen und Kräfte. Kraft und Momentmessungen aus den internen Sensoren unterliegen einem Rauschen, das die Klassifikation erschwert. Die Positionen der Gelenke können dagegen sehr präzise gemessen werden. Daher werden neben den Kräften und Momenten auch TCP Pose und Roboterkonfiguration berücksichtigt.

Um die Generalisierung zwischen unterschiedlichen Ausführungen zu vereinfachen, werden absolute Werte als Merkmale vermieden. Stattdessen werden Differenzen zu erwarteten Werten aus den Manipulationsplanern überwacht. So sagt die absolute Armkonfiguration im Allgemeinen sehr wenig über den Zustand des Systems aus. Erreicht sie jedoch ihren kommandierten Wert nicht, so kann dies durch ein kollidierendes Hindernis verursacht sein.

Es wird angenommen, dass jeder gewollte Kontakt zur Umwelt durch den Greifer des Roboters entsteht. Er ist starr mit dem Manipulator im TCP verbunden. Damit wird die Pose des Greifers einzig durch den TCP vorgegeben. Da die korrekte Pose des Greifers für die Manipulation entscheidend ist, wird die Abweichung der gemessenen TCP Pose von der kommandierten überwacht.

Name	Beschreibung	Dimension
$\Delta\mathbf{x_{TCP}}$	Differenz zwischen kommandierter und gemessene TCP Pose	6
$\lvert\Delta x_{TCP}\rvert$	Distanz der kommandierten zur gemessenen TCP Position	1
$\Delta\mathbf{F_{TCP}}$	Differenz TCP zwischen gemessener Kraft bzw. Moment und erwarteter	6
$\lvert\Delta F_{TCP}\rvert$	Betrag der Differenz zwischen gemessener und erwarteter Kraft am TCP	1
$\Delta\theta$	Differenz zwischen kommandierter und gemessener Greiferkonfiguration	12
$\Delta\tau$	Differenz der erwarteten und gemessenen Greifermomente	12
$\lvert\Delta\mathbf{x_{tip}}\rvert$	Distanz zwischen Finger- und Daumenspitze	1
	Summe	41

Tab. 7.1.: Der zur Ausführungsüberwachung verwendete Merkmalsvektor

Aufgrund ähnlicher Überlegungen wird die Differenz zwischen erwarteten und gemessenen Kräften und Momenten im TCP berücksichtigt. Greifen und Transportieren von Objekten sind die wichtigsten Aktionen für die Manipulation. Ist ein Objekt gegriffen, übt es über die Hand Kräfte und Momente auf den Arm aus. Diese sollen für die Überwachung verwendet werden können. Dabei wird sowohl der sechsdimensionale Vektor, als auch sein Betrag verwendet.

Während man beim Arm davon ausgehen kann, dass nur sein Ende, der TCP, für den Erfolg einer Aktion verantwortlich ist, ist diese Annahme für die Finger des Greifers nicht gültig. Dort kann jedes Fingersegment Kontakt mit dem Objekt herstellen und damit zur Manipulation beitragen. Daher wird die vollständige Konfiguration des Greifers berücksichtigt. Die Differenz der gemessenen und kommandierten Konfiguration wird ebenso verwendet, wie die Differenz der gemessenen und kommandierten Momente.

Rutscht ein gegriffenes Objekt aus dem Greifer, so ist die Impedanzregelung weiterhin parametriert, eine Kraft auf das Objekt auszuüben. Die Finger werden daher weiter geschlossen, bis ihre virtuelle Feder im Gleichgewicht ist. Bei kleinen Objekten kann dies dazu führen, dass opponierende Finger kollidieren. Die Finger üben dann gegenseitig eine Kraft aufeinander aus, die es unmöglich macht, das Fehlen des Objektes an einer fehlenden Kraft zu erkennen. Aus diesem Grund wird das Merkmal "Distanz zwischen Finger- und Daumenspitze" eingeführt. Anhand dieser können Kollisionen zwischen den Fingen und damit Griffe ins Leere erkannt werden.

Die überwachten Merkmale für einen bimanuellen Serviceroboter sind in Tabelle 7.1 zusammengefasst. Insgesamt hat der Merkmalsraum 41 Dimensionen für jeden Arm des Demonstrators mit je 20 Freiheitsgraden.

7.2.3.2. Lernverfahren

Aus der Menge der vorhandenen Kanäle müssen die relevanten und ihre zulässigen Wertebereiche bestimmt werden. Dafür wird ein Lernverfahren verwendet. Das Wissen wird nicht explizit erzeugt. Stattdessen wird wird ein Datenpunkt zu einem Zeitpunkt klassifiziert. Das Verwenden einzelner Datenpunkte entspricht zum einen den Voraussetzungen typischer Lernverfahren wie Bayes'sche Netze, k-Nächster Nachbar oder Neuronale Netze (Duda u. a., 2001). Zum anderen ist durch diesen Ansatz eine große Menge von Trainingsdaten mit geringen Aufwand zu erzeugen. Dazu werden die Datenpunkte einer Sequenz aufgezeichnet und einzeln als Trainingsbeispiele genutzt. Weiter erlaubt die Nutzung einzelner Punkte im Gegensatz zu Klassifikation einer Sequenz die Anwendung in Echtzeit.

Die Datenpunkte werden in die Klassen „erfolgreich" und „fehlgeschlagen" klassifiziert. Im Gegensatz zu klassischen PdV Lernansätzen demonstriert der Roboter die Ausführung einer Aktion dem Benutzer. Der Benutzer bewertet die Ausführung als „erfolgreich" oder als „fehlgeschlagen". Dies ist im laufenden Betrieb möglich und kann z.B. durch Auswerten des „Not-Aus"-Systems geschehen. Auch muss der Benutzer kein Experte sein, da kein Wissen über die Implementierung der Aktionsausführung benötigt wird, sondern auf die intuitive Vorstellung der Aufgabenausführung zurückgegriffen wird.

Der verwendete Merkmalsraum enthält, abhängig von der ausgeführten Aktion größtenteils irrelevante Merkmale, die nicht sinnvoll zur Klassifikation beitragen. Aus Sicht des Klassifikators enthalten diese Kanäle ausschließlich Rauschen. In so einer Konstellation besteht die Gefahr, dass der Klassifikator im Rauschen Strukturen entdeckt, die ebenfalls konsistent mit der gesuchten Klassifikation sind. Diese führen zu einer guten Klassifikation der Trainingsdaten, jedoch zu schlechten Ergebnissen bei neuen Datenpunkten. Der Effekt wird Überanpassung genannt. Aus diesem Grund wird in dieser Arbeit eine Support Vector Maschine (SVM) verwendet. Dieses Lernverfahren ist besonders robust gegenüber Überanpassung (s. Abschnitt 2.6). Weiter kann es sehr gut mit hochdimensionalen Datenräumen und irrelevanten Merkmalen umgehen. Der „Fluch der Dimensionalität" führt dazu, das bei einem 41 dimensionalen Datenraum sehr viele Trainingsbeispiele benötigt würden. Eine SVM reduziert den Bedarf auf eine Anzahl Beispiele, die nur mit der Anzahl der relevanten Dimensionen das Merkmalsraumes steigt.

Als Kernel der SVM kommen Radiale Basis Funktionen zum Einsatz. Als Implementierung der SVM wird auf die LIBSVM (Chang u. Lin, 2001) zurückgegriffen. Neben der Optimierung der Trennebene implementiert diese ebenfalls eine Optimierung der Parameter der SVM aufgrund der gegebenen Trainingsdaten.

Für jede atomare Aktion wird eine SVM erzeugt. Dazu führt der Roboter eine Handlungssequenz aus, die aus mehreren (atomaren) Aktionen besteht. Der Benutzer muss lediglich das Fehlschlagen der Aktion anzeigen. Das System kann automatisch die Datenpunkte den

Aktionen zuordnen. Wird ein Fehler angezeigt, so ist der Zeitpunkt des Fehlschlages möglichst genau zu bestimmen. Dazu kann dem Benutzer ein vom Roboter aufgezeichnetes Video angezeigt werden, in dem er den Zeitpunkt des Fehlers genau markieren kann. Zusätzlich wird um den Zeitpunkt ein Zeitfenster der Aufnahme ausgeblendet, um keine falsch zugeordneten Datenpunkte zu erzeugen.

Es gibt Aktionen, für die es keine zeitinvarianten Kanäle gibt. Z. Bsp. ist es für das Zugreifen nicht möglich, einen solchen zu identifizieren. Der korrekte Anfangszustand ohne gegriffenes Objekt entspricht am Ende der Aktion einem Fehler. Als Gegenmaßnahme lässt sich das Verfahren erweitern. Der Endzeitpunkt einer Aktion ist leicht zu identifizieren und hat meist definierte Anforderungen. Man kann daher weiter SVMs nur zur Klassifikation dieses Punktes trainieren. Damit kann eine Erfolgsbewertung über die symbolisch festgelegten Eigenschaften hinaus durchgeführt werden. Nachteilhaft ist allerdings, dass für diese Punkte weniger Trainingsdaten aus den Demonstrationen gewonnen werden können. Auch kann damit für die Zeit davor keine Aussage über Fehler in Echtzeit gemacht werden. Diese Erweiterung wurde daher nicht weiter evaluiert.

7.3. Fazit

In diesem Abschnitt der Arbeit wurde auf die Konkretisierung symbolischer Aktionen eingegangen. Ihre Parametrierung erfolgt zu großen Teilen bereits im Symbolischen. Dort werden Aktionssequenzen auf Flexible Programme abgebildet. Der Konkretisierung bleibt damit im Wesentlichen die Zuordnung zu benötigten Planern und Ausführungskomponenten und die Abbildung der Parameter in eine geeignete Struktur. Das Wissen dazu wird in den Flexiblen Komponenten gespeichert.

Um auf ungeplante Abweichungen bei der Ausführung reagieren zu können, wird eine Komponente zur Ausführungsüberwachung benötigt. Dazu wurde ein auf überwachten Lernen basierter Ansatz entwickelt. Der Benutzer annotiert Vorführungen einer Aktionssequenz. Basierend auf den Sensorwerten der internen Robotersensorik wird eine Support Vector Machine trainiert. Diese ist in der Lage, Fehler in der Ausführung in Echtzeit zu erkennen. Aufgrund ihrer Ergebnisse kann eine Neuplanung ausgelöst werden.

8. Experimentelle Evaluation

In diesem Kapitel wird der vorgestellte Ansatz zur szenenabhängigen Online-Adaption von Handlungssequenzen experimentell evaluiert. Dazu wurde das entwickelt System implementiert. Die erstellte Software ist auf unterschiedlichen Robotern verwendbar. Die zur Evaluation eingesetzten Roboter werden in Abschnitt 8.1 vorgestellt.

In Abschnitt 8.2 werden die einzelnen Komponenten des Systems in definierter Umgebung unabhängig voneinander betrachtet. Im darauf folgenden Abschnitt 8.3 steht dagegen das Zusammenspiel der Komponenten im Blickpunkt. Dazu wird das System zur Lösung von Manipulationsaufgaben unterschiedlicher Komplexität eingesetzt.

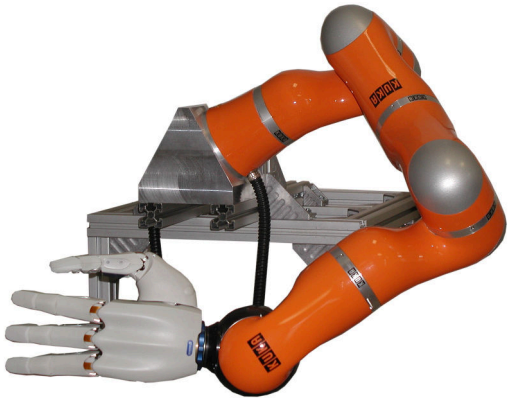
8.1. Robotersysteme

Das entworfene System nutzt eine abstrakte Modellierung des ausführenden Roboters. Die vorgestellte Planungsdomäne und Szenenmodellierung kann daher verwendet werden, um unterschiedliche Roboter zu steuern. Dies ist auf den im Folgenden vorgestellten Demonstratoren umgesetzt worden. Sie wurden im Rahmen öffentlich geförderter Projekte entwickelt. Besonders zu erwähnen sind das BMBF geförderte DESIRE Projekt[1], das EU Projekt DEXMART[2] und das BMWi finanzierte ViEMA[3] Projekt. Die Roboter sind aus einem kontinuierlichen Weiterentwicklungsprozess entstanden und werden hier in historischer Reihenfolge präsentiert.

Der einarmige FZI / DESIRE Demonstrator besteht aus einem KUKA Leichtbauarm III (LBR) (Bischoff u. Kurth, 2010) und einer Schunk Anthropomorphen Hand (SAH) (Liu u. a., 2007). Die Aktuatoren sind auf einem Gerüst befestigt, sodass sie auf üblicher Tischhöhe arbeiten können.

Der KUKA LBR (III und IV) ist ein Leichtbauroboter mit 7 Freiheitsgraden. Er wiegt 15 kg und kann eine Last von 7 kg handhaben. Seine Gelenke sind mit Momentensensoren ausgestattet. Seine Regelung erfolgt mit einer Frequenz von 1 kHz. Dies ermöglicht eine nachgiebige Regelung, die als Impedanzregelung umgesetzt ist, sodass sich der Roboter wie ein Feder-Dämpfer-

[1] http://www.service-robotik-initiative.de/
[2] http://www.dexmart.eu/
[3] http://viema.org



(a)

(b)

(c)

(d)

(e)

Abb. 8.1.: Abbildung der unterschiedlichen Robotersysteme, auf denen die vorgestellten Verfahren ausgeführt wurden. In (a) ist eine frühe, einarmige Version des FZI Demonstrators aus dem DESIRE Projekt zu sehen. Sie besteht aus einem KUKA LBR und einer Schunk Anthropomorphen Hand. In (b) wurde das System mit einer mobilen Plattform und einem zweiten LBR Arm erweitert. Es verwendet die Schunk Dextrous Hand. Abb. (c) stellt den zweiarmigen Demonstrator des FZI dar, der zwei LBR mit zwei DLR/HIT Händen vereinigt. Im Bild ist er in der AISROB Konfiguration dargestellt. In Abschnitt 8.1(d) ist das System mit einer KUKA omniRob Plattform erweitert worden, die Mobilität erlaubt. Abb. (e) zeigt die Viema Zelle, bei der ein Denso Arm auf einem Drehteller montiert ist.

(a) (b)

Abb. 8.2.: Abbildung der zur Hindernissmodellierung verwendeten Tiefenbildsensoren. (a): Swissranger 4000 der Firma Mesa Imaging, der sowohl auf dem DESIRE Demonstrator als auch auf dem zweiarmigen FZI Demonstrator eingesetzt wurde. (b): Kinect Sensor der Firma Microsoft.

System verhält. Die Steuerung des Roboters erlaubt dabei die Berücksichtigung von Lasten und den Ausgleich der wirkenden Schwerkraft.

Die SAH wurde vom Harbin Institute of Technology (HIT) in China, dem Institut für Mechatronik der Deutschen Gesellschaft für Luft- und Raumfahrt (DLR) und der Firma Schunk entwickelt. Eine Abbildung der Hand ist in Abb. 8.1(a) sowie Abb. 8.3 zu sehen. Sie basiert auf der DLR-Hand II und hat dreizehn Freiheitsgrade. Die Hand ist aus vier gleichen Fingermodulen aufgebaut. Jedes Modul hat vier Gelenke; das mediale und distale sind mechanisch gekoppelt, sodass sich drei Freiheitsgrade pro Finger ergeben. An jedem Freiheitsgrad befindet sich ein Momentensensor. Ein Finger ist den drei anderen gegenüber angeordnet und dient als Daumen. Er hat einen weiteren Freiheitsgrad an seiner Basis. Zur Regelung der Hand kommt ein positionsbasierter sowie ein impedanzbasierter Modus zum Einsatz .

Im Rahmen des DESIRE Projektes entstand aus dem im vorigen beschriebenen, ortsfesten Aufbau ein zweiarmiger mobiler Service Roboter. Dieser ist in Abb. 8.1(b) zu sehen. Die Mobilität wird durch die omnidirektionale MPO-700 Platform der Firma Neobotix erreicht. Sie beinhaltet Laserscanner zur Hindernisvermeidung. Für die Manipulation werden die Scanner nicht verwendet. Auf der Plattform sind an einem Gerüst zwei KUKA LBR montiert, die jeweils eine Schunk Dexterous Hand (SDH) als Endeffektor besitzen.

An einer Schwenk-Neige-Einheit (engl. pan-tilt-unit, PTU) ist ein Sensor Kopf angebracht, mit dem der Roboter seine Umgebung wahrnimmt. Er besteht wiederum aus einer 3D Time-of-Flight (TOF) Kamera SwissRanger SR3000 der Firma Mesa Imaging (dargestellt in Abb. 8.2(a)) sowie einem Stereokamerasystem aus zwei AVT PIKE 145C Kameras.

Der SwissRanger erzeugt ein Tiefenbild der Umgebung mit einer Auflösung von 176 x 144 Punkten und einer Bildrate von 25 fps. In einer Entfernung von 1,5 m kann der Sensor damit Würfel von etwa einem Zentimeter Kantenlänge auflösen. Der Sensor wird zur Perzeption von

Abb. 8.3.: Abbildung der zur Evaluation verwendeten Greifer. Von Links nach rechts: Schunk Anthropomorphe Vierfingerhand (SAH), Röhm Zweibackengreifer REPG 30 und DLR/HIT Fünffingerhand.

nicht modellierten Hindernissen eingesetzt; das Verfahren dazu wird von (Kühnle u. a., 2009) beschrieben. Das Stereokamerasystem wird zur Lokalisierung bekannter Objekte verwendet. Es kommt das Verfahren aus (Grundmann u. a., 2010) zum Einsatz.

Eine erweiterte Variante des FZI Demonstrators ist in Abb. 8.1(c) dargestellt. Die abgebildete Konfiguration besteht aus einem ortsfesten Aufbau mit zwei Kuka LBR und zwei DLR/HIT Fünf-Fingerhänden. Er kann jedoch ebenfalls mit der SAH, SDH oder Zweibackengreifern verwendet werden. Der verwendete Sensorkopf ist bauglich mit dem des mobilen Demonstrators des DESIRE Projektes, lediglich die Befestigung ist in diesem System fix. Die DLR/HIT Fünffingerhand besitzt fünf Finger mit je vier Gelenken und drei Freiheitsgraden. Wie bei der SAH sind die vorderen Gelenke jedes Fingers mechanisch gekoppelt. Es fehlt jedoch der zusätzliche Freiheitsgrad des Daumens. Insgesamt hat die Hand fünfzehn Freiheitsgrade. Auch sie ist mit einer Impedanzregelung nachgiebig regelbar. Alle an dem Roboter eingesetzten Greifer sind in Abb. 8.3 zu sehen.

Die aktuellste Version des Demonstrators ist in Abb. 8.1(d) abgebildet. Arm und Handkonfiguration sind gegenüber dem Vorgänger unverändert. Hinzugekommen ist die Mobilität des Demonstrators, die durch eine KUKA omniRob Plattform erreicht wird. Weiter kommt ein Kinect RGB-D Sensor zum Einsatz, der Tiefenbilder durch Musterprojektion erzeugt. Dieser wird als Ersatz für den TOF Sensor zum Erzeugen von Hindernismodellen verwendet. Die Auflösung der Tiefenbilder beträgt 640 x 480 Bildpunkte. Der Sensor ist in Abb. 8.2(b) abgebildet.

Ein weiteres System, auf dem die vorgestellte Methodik Umgesetzt wurde, ist die im ViEMA Projekt aufgebaute Roboterzelle, wie sie in Abb. 8.1(e) dargestellt ist. Die Zelle besteht aus einem sechsachsigen DENSO Arm vom Typ New VS-087. Dieser hat eine Reichweite maximale von 900 mm. Um den erreichbaren Arbeitsraum des Arms zu vergrößern, ist dieser auf einem Drehteller montiert.

(a) (b) (c)

Abb. 8.4.: Das Sauerkrautdosen Objekt in drei verschiedenen stabilen Lagen visualisiert. Die linke Lage entsteht direkt aus einer erkannten Ebene der konvexen Hülle. Die beiden rechten Lagen entstammen dem abschließenden Abtastschritt. Insgesamt werden 30 Lagen für das Objekt erzeugt.

Die verwendeten Arme werden in ihrer Standardkonfiguration über Steuerungsrechner des Herstellers betrieben. Diesen wird eine Konfiguration, bzw. eine Liste von Konfigurationen übergeben, die von der Robersteuerung mit gegebenen maximalen Geschwindigkeiten und Beschleunigungen interpoliert und abgefahren wird. Für die vorliegende Arbeit ist diese Art der Ansteuerung nicht geeignet, da sie nicht die synchrone Ansteuerung mehrerer Komponenten erlaubt. Stattdessen wird für diese Arbeit eine Echtzeitschnittstelle verwendet. Sie erlaubt es, den Robotern in Takten bis zu 1 kHz Sollkonfiguartionen über das UDP Protokoll zu kommandieren. Dadurch ist es möglich, die Interpolation in höhere Ebenen zu ziehen, die mehrere Komponenten berücksichtigen können.

Als Steuerungsrechner für das System kamen unterschiedliche Systeme zum Einsatz. Wo diese zum Vergleich von Laufzeiten relevant sind, werden sie in den folgenden Abschnitten explizit beschrieben.

8.2. Evaluation der Systemkomponenten

In diesem Abschnitt werden die Komponenten zur Arbeitsraummodellierung, Erzeugung mechanischer Relationen und Ausführungsüberwachung unabhängig voneinander betrachtet.

8.2.1. Arbeitsraumbeschreibung

Als Bewertungsfunktion zur Auswahl beliebiger Orte für die symbolische Planung wurde in dieser Arbeit das Maß der Greifbarkeit eingeführt. Die Bestimmung der Greifbarkeit basiert auf stabilen Ebenen des Objektes, die mit dem Verfahren aus (Xue u. a., 2008b) bestimmt werden.

Ein Ergebnis des Verfahrens für ein zylinderförmiges Objekt ist in Abb. 8.4 dargestellt. Die Lage in Abb. 8.4(a) und die nicht dargestellte Lage auf der gegenüberliegenden Seite stammen aus dem auf der konvexen Hülle basierenden Teil des Verfahren. Die Lagen in Abb. 8.4(b) und Abb. 8.4(c) stammen aus dem abschließenden Abtastschritt, der noch 28 weitere Lagen auf der Seite des Zylinders erzeugt. Das Verfahren erzeugt die erwarten Objektlagen, auf denen die Greifbarkeit berechnet wird.

In Abb. 8.5 sind Greifbarkeitskarten für das Sauerkrautdosenobjekt visualisiert. Sie gehören zu den drei in in Abb. 8.4 gezeigten stabilen Lagen des Objektes und beschreiben den rechten Arm des Roboters. Der Arm befindet sich an folgender Pose im Raum x: 348 mm, y: -41 mm, 1292 mm, Rollwinkel: -1.053, Nickwinkel: -0.021, Gierwinkel: 3.117. Dabei steht der Roboter vor dem Tisch, in Abb. 8.5 am unteren Rand des Bildes. Der Mittelpunkt des Tisches befindet sich bei x: 800 mm, y: 0 mm, z: 900 mm. Die Anordnung mit Roboter ist in Abb. 8.7 dargestellt.

Für die stehende Dose in Abb. 8.4(a) fällt die Greifbarkeit in (Abb. 8.5(a)) am Rand flacher ab, da Griffe von der Seite (Abb. 2.9(a)) flexibel angewandt werden können. Für die liegende Richtung sind nur Griffe von den Kopfseiten, von oben und senkrecht zur Achse des Zylinders bekannt. Die Diagramme in Abb. 8.5(b) und Abb. 8.5(c) für die liegenden Lagen sind sehr ähnlich, dies spiegelt die Rotationssymmetrie des Objektes wider. Die Greifbarkeitskarten der verbleibenden 26 erzeugten Lagen des Objektes sind ebenfalls sehr ähnlich zu den beiden vorgestellten und werden daher nicht abgedruckt.

Abb. 8.6 zeigt die Greifbarkeitskarten für das Ceylontee Objekt. Das Objekt selbst, sowie eine Projektion seiner Greifbarkeitskarte in eine visualisierte Umgebung des Roboters ist in Abb. 8.7 zu sehen. Die Greifbarkeitskarten sind für den linken und rechten Arm dargestellt. Sie zeigen eine große Ähnlichkeit, sind jedoch nicht völlig symetrisch. Dies ist zum einen drauf zurück zu führen, dass die Arme nicht gespiegelt am Roboter montiert sind, sondern um 180° rotiert. Dies führt zusammen mit den Gelenkwinkelanschlägen zu unerschiedlichen möglichen Armkonfigurationen. Zum anderen ist auch die Basis der Arme nicht exakt symmetrisch modelliert. Ihre Lage wird aus einer Kalibrierung bestimmt, die für beide Arme kleine Abweichungen der Lage feststellt. Zusammen mit der Diskretisierung führt dies zu Unterschieden in den beiden Karten.

Abb. 8.6(c) zeigt die Fusion der beiden Karten mittels der Minimumfunktion. Diese kann verwendet werden, um einen Ort für eine Übergabe zu bestimmen. Wie gewünscht liefert die Fusion einen Bereich in der Mitte des Arbeitraumes, der von beiden Armen zu erreichen ist.

Ein typisches zu beobachtendes Merkmal der meisten Greifbarkeitskarten ist die bogenförmige Grenze zwischen dem greifbaren Bereich auf der dem Arm zugewandten Seite und dem nicht greifbaren Bereich am anderen Ende des Tisches, zu sehen sowohl in Abb. 8.5 als auch in Abb. 8.7. Sie entsteht durch die Begrenzung der maximalen Entfernung eines greifbaren Objektes durch die Armlänge. Es ist auch zu erkennen, dass die Greifbarkeit neben der Entfernung

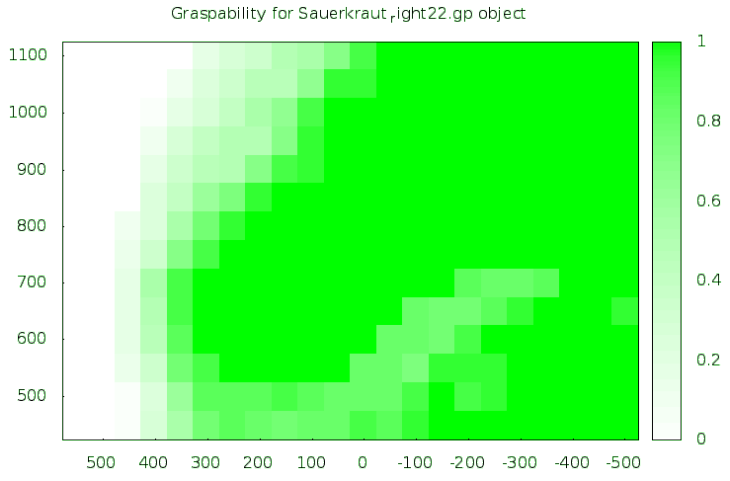
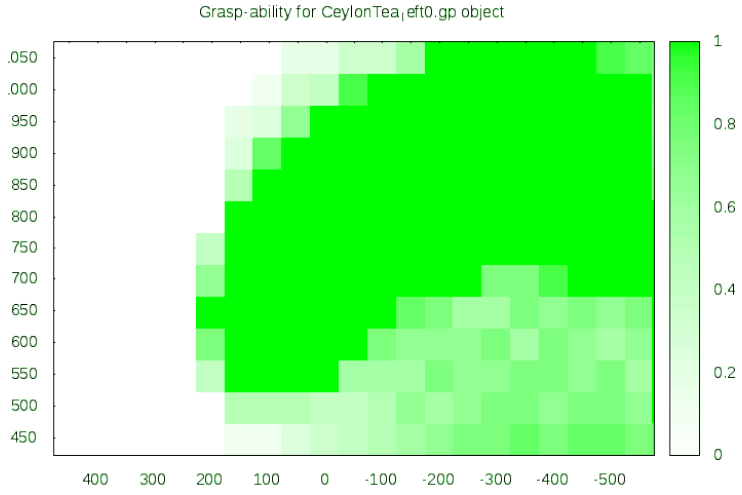
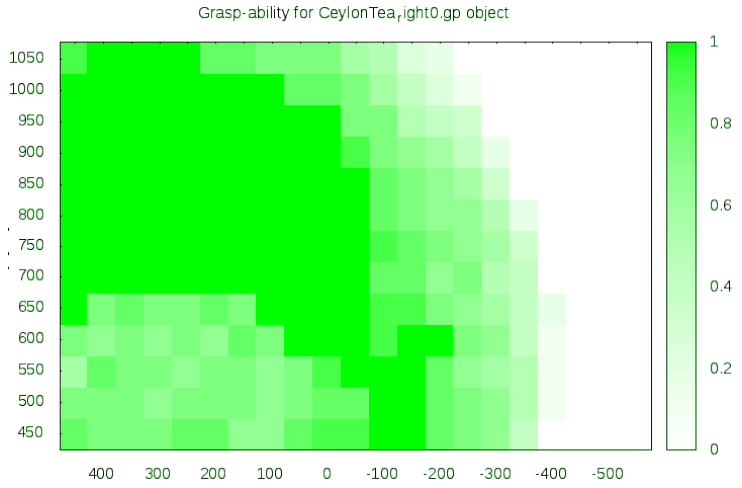


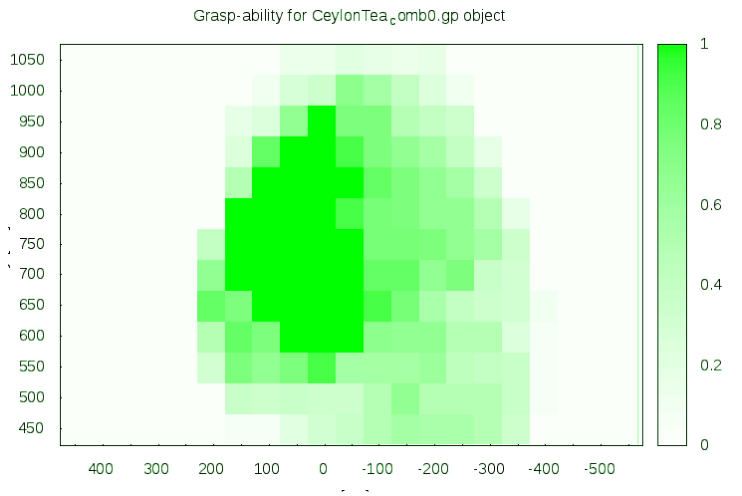
Abb. 8.5.: Greifbarkeit des Saurkrautdosen Objektes. Z wird mit 80 cm angenommen. In der Visualisierung stünde der Manipulator am unteren Bildrand. Drei stabile Lagen sind visualisiert. Sie sind in Abb. 8.4 dargestellt. Die Übergänge zwischen den Bereichen sind weich, es gibt keine Ausreißer. Für alle Lagen ergeben sich ähnliche Bilder, Unterschiede liegen im Detail.



(a)



(b)



(c)

Abb. 8.6.: Greifbarkeit des Ceylontee Objektes. Z wird mit 80 cm angenommen. In der Visualisierung stünde der Manipulator am unteren Bildrand. In (a) ist eine Greifbarkeitskarte für den linken Arm zu sehen, in (b) das Gegenstück für das selbe Objekt in der selben Lage für den rechten Arm. (c) zeigt die Fusion mittels der Minimumfunktion für eine Übergabeaktion.

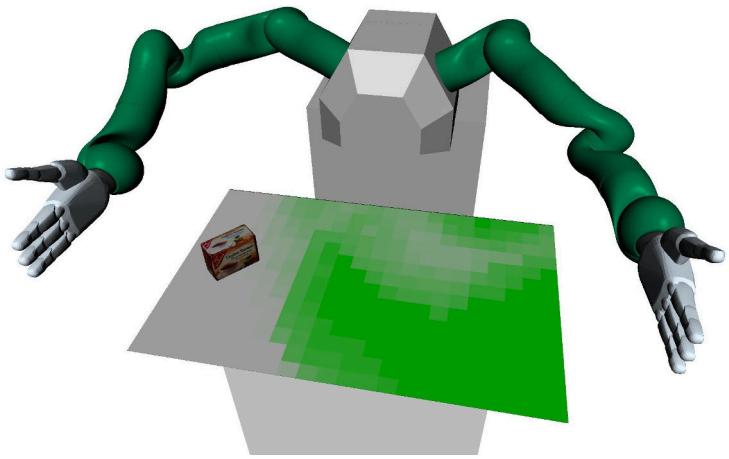


Abb. 8.7.: Greifbarkeit des Ceylon Tee Objektes in der dargestellten Lage für den linken Arm auf den angenommenen Tisch projiziert.

Objekt	# Griffe	# Ebenen	# Einräge	Zeit zur Erstellung [h]
Ceylon Tea	24	6	1848	1 h
Sauerkraut	59	23	7084	6 h

Tab. 8.1.: Zeitverbrauch zur Erstellung der Greifbarkeitskarte sowie ihre Größe.

noch von weiteren Einflüssen der Kinematik abhängt. In allen Karten gibt es Bereiche, die nicht erreichbar sind, obwohl sowohl davor als auch dahinter erreichbare Orte liegen.

Eine allgemeine Beobachtung ist, dass die Greifbarkeit ein annähernd glattes Maß darstellt, d. h. die Ränder fallen kontinuierlich ab und es sind keine Ausreißer in den abgebildeten Visualisierungen zu sehen. Dies ist ein Indiz dafür, dass die gewählte Abtastdichte mit Quadraten von 5 cm Kantenlänge geeignet ist, die Greifbarkeit wiederzugeben.

In den Abbildungen 8.6 ist weiter zu erkennen, dass es in der Greifbarkeitskarte Gebiete gibt, in denen eine Art Rauschen zu sichtbar ist. Diese sind nahe der Schulter des Roboters (in den Bildern unten rechts bzw. links). Sie sind darauf zurückzuführen, dass der Arm in diesem Gebiet nahe an den Gelenkwinkelbegrenzungen arbeitet.

In Tabelle 8.1 wird der Zeitverbrauch zur Erstellung von Greifbarkeitskarten auf einem Intel Core i7 mit 3GB RAM betrachtet. Im wesentlichen hängt der Zeitbedarf linear von der Anzahl der Griffe für das Objekt, der Anzahl an stabilen Lagen und der Anzahl der diskretisierten Posen ab. Ein Objekt in der verwendeten Objektdatenbank hat im Mittel zwölf stabile Lagen und das Berechnen seiner Greifbarkeit benötigt ca. 2 h. Bei 100 Objektmodellen ist dies ein aufwändiger Prozess. Allerdings sind die Greifbarkeiten für die einzelnen Objekte und Ebenen völlig unabhängig voneinander. Es ist daher möglich, die Berechnung auf mehrere Prozesse und unterschiedliche Rechner zu verteilen. Zur Berechnung der vollständigen Datenbank wurden fünfzehn Rechner unterschiedlicher Konfigurationen verwendet (Minimum Athlon X2, 2

(a) (b)

Abb. 8.8.: Beispiel für einen erzeugten Szenengraph. Die Szene ist in (a) gezeigt. Der Szenengraph ist in (b) dargestellt. Für den Tisch werden keine bewegungsbasierten Relationen erzeugt. Die „starr" Relation ist der Übersichtlichkeit halber vernachlässigt.

GB RAM, Maximum Intel Core i7, 3GB RAM). Auf jedem Rechner liefen zwei Prozesse für unterschiedliche Objekte. Der Vorgang war nach spätestens 10 Stunden abgeschlossen.

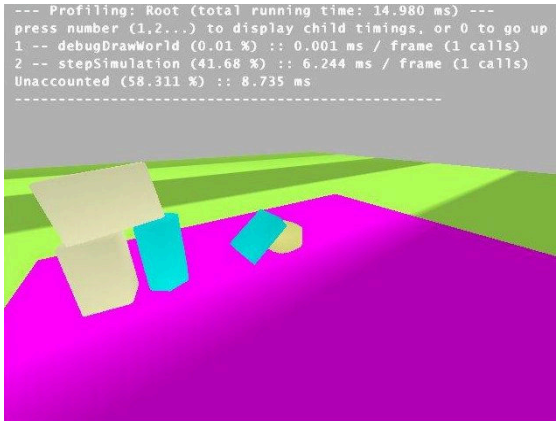
Diese Zeiten haben durchaus Relevanz für die praktische Entwicklung des Verfahrens. Für die Verwendung im Planer spielen sie jedoch keine Rolle, da die Karten im Voraus offline berechnet werden können. Die Zugriffszeiten, um sie online auszuwerten, sind vernachlässigbar.

8.2.2. Erzeugung mechanischer Relationen zur Szenenbeschreibung

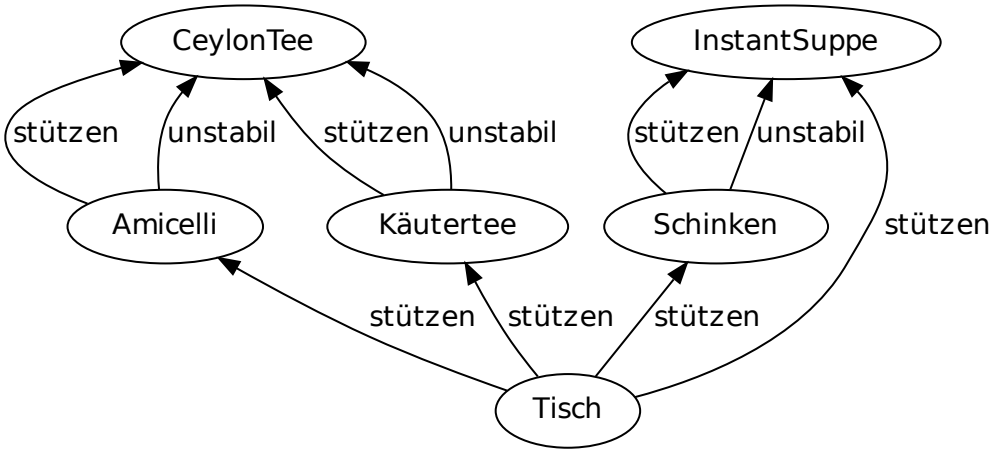
Eine einfache Szene ist in Abb. 8.8(a) dargestellt. Auf diese wird das Verfahren zum automatischen Erzeugen mechanischer Relationen angewandt. Dabei werden neben der Berechnung der Kräfte zwischen den Objekten für jedes Objekt zehn Bewegungstests ausgeführt. Die Objekte werden jeweils entlang der positiven und negativen X- und Y-Achse sowie der positiven Z-Achse um eine Distanz von 2 cm und 5 cm bewegt. Dabei wird eine Geschwindigkeit von 0.05 und 0.25 $\frac{\mathrm{m}}{\mathrm{s}}$ simuliert.

Die einzelnen Tests liefern unterschiedliche Ergebnisse. Die schnellen Tests bewerten die Konfiguration als unstabil, drei der langsamen Tests dagegen führen zu einer stabilen Relation. Insgesamt führt der Mehrheitsentscheid damit zu einer „Unstabil" Relation. Händische Experimente mit einem analog aufgebauten Stapel stützen das Ergebnis. Für den Menschen ist es kein Problem, die Objekte gemeinsam zu bewegen. Andererseits genügt ein Ruck, um das obere Objekt zum Fallen zu bringen. Für die Manipulation durch den Roboter ist der Stapel in dieser Form daher ungeeignet. Die unterschiedlichen Testergebnisse zeigen, dass eine geeignete Pa-

(a)



(b)



(c)

Abb. 8.9.: Erzeugter Szenengraph. Die Szene ist in (a) gezeigt, (b) zeigt die Darstellung in der Physiksimulation. Der Szenengraph ist in (c) dargestellt. Für den Tisch werden keine dynamischen Beziehungen erzeugt, da dieser durch den Roboter nicht zu bewegen ist. Die Beziehungen der getragenen Objekte „Ceylontee" und „ Instant Suppe" zu ihren tragenden Objekten sind korrekt erkannt. Sie sind instabil, da das Entfernen eines einzelnen Trägers zum Fallen des Objektes führt.

Trainierte Objekte	1	5
Trainierte Griffe	2	13
Erzeugte Trainingsdatenpunkte	244	795
Korrekt klassifizierte Trainingsdaten	100 %	100 %
Korrekt klassifizierte Testdaten (einfach)	45 %	99,8 %
Korrekt klassifizierte Datenpunkte mit Störung		93,4 %
Erzeugte Anzahl Stützvektoren	43	159
Trainingsdauer der SVM	3,7 s	16,9 s
Demonstrationszeit	10 min	50 min

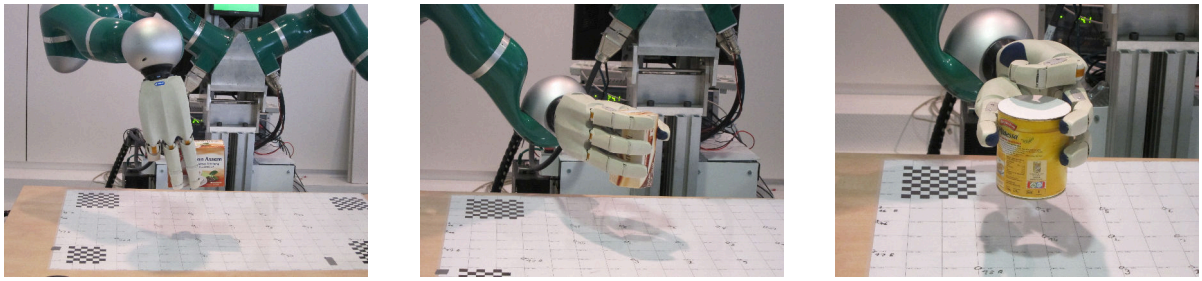
Tab. 8.2.: Ergebnisse der SVM zur Ausführungsüberwachung für zwei Trainingsmengen. Das Objekt wird durch den Roboter gegriffen und transportiert. Die Aktion schlägt fehl, wenn das Objekt die Hand während der Bewegung verlässt, oder es zu einer Kollision mit einem Hindernis kommt.

rameterwahl für die erfolgreiche Anwendung des Verfahrens notwendig ist. Der resultierende Szenengraph ist in Abb. 8.8 dargestellt.

Die Experimente zur Evaluation der mechanischen Szenenrelationen in diesem Abschnitt wurden auf einem PC mit einer Intel(R) Core(TM)2 Duo T7800 CPU durchgeführt, die auf 2.60 GHz getaktet war und zugriff auf 4 GB RAM hatte. Zur Berechnung des Szenengraphs waren aus Abb. 8.8 wurden 11 Sekunden Rechenzeit benötigt.

In einem zweiten Experiment wird das Verfahren auf eine komplexere Szene angewandt. Sie ist in Abb. 8.9(a) abgebildet, ihre Visualisierung in der physikalischen Simulation ist in Abb. 8.9(b) zu sehen. Es sind fünf Objekte in zwei Gruppen vorhanden. In beiden Gruppen ist die besondere Herausforderung, dass die Schwerpunkte der gestützten Objekte nicht über den tragenden liegen. Eine solche Szene ist für einen rein geometriebasierten Ansatz kaum korrekt zu interpretieren.

Der vorgestellte Ansatz erzeugt den in Abb. 8.9(c) visualisierten Szenengraphen, der mit der intuitiven Wahrnehmung der Szene übereinstimmt. Es werden die selben Tests durchgeführt wie im ersten Versuch. Dazu werden insgesamt 1052 Simulationsschritte berechnet. Auf dem Versuchsrechner benötigt der Prozess 33 Sekunden. Von diesen wird ein vernachlässigbar kleiner Anteil für das Berechnen der statischen Kräfte zwischen den Objekten (< 0,5 s) benötigt, fast die gesamte Zeit wird mit den Bewegungstests verbracht. Da die einzelnen Bewegungstests voneinander unabhängig sind, kann davon ausgegangen werden, dass hier durch Parallelisierung noch ein großes Potential zur Beschleunigung besteht. Auch Hardware bzw. GPU unterstützte Simulationsansätze könnten dazu einen weiteren Beitrag leisten.



(a) (b) (c)

Abb. 8.10.: Trainingsphase der Ausführungsüberwachung: Der Demonstrator greift erfolgreich drei unterschiedliche Objekte. Die Objekte unterscheiden sich in Form und Größe. Es werden Griffe unterschiedlicher Art mit verschiedenen Anrückrichtungen erzeugt. Aufgrund der unterschiedlichen Griffe wirken unterschiedliche Kräfte und Momente. So ist in Bild (a) der dritte Finger nicht am Griff beteiligt, in (c) dagegen der erste. Zum Erzeugen negativer Beispiele werden die Objekte entfernt.

8.2.3. Ausführungsüberwachung

Um die Ausführungsüberwachung unabhängig von anderen Komponenten des Systems durchzuführen, wird in diesem Versuch die Überwachung des Greifens eines einzelnen Objektes und des anschließenden Transportes betrachtet. Die Aktion ist dabei vorgegeben, es findet keine symbolische Planung statt.

Die Ausführungsüberwachung basiert auf Zustandswerten der Impedanzregelung, entsprechende Sensoren müssen auf Hardwareseite verfügbar sein. In diesem Versuch wird die Schunk Anthropomorphe Vierfingerhand eingesetzt. Als Arm kommt der KUKA LBR 4 zum Einsatz. Die Berechnungen werden auf einer Intel(R) Core i7 CPU 920 mit 2.67 GHz und 3 Gb RAM ausgeführt.

Für den Versuch wurden fünf Haushaltsobjekte aus der Karlsruher Objektdatenbank ausgewählt. Die Auswahl richtet sich dabei nach den Fähigkeiten der verwendeten Hand. Sie bestimmt Gewicht und Größe der Objekte. Bei der Form wird eine möglichst große Vielfalt angestrebt, um verschiedenartige Griffe zu erhalten. Bei den Objekten handelt es sich um „Amicelli", „CeylonTea", „DanishHam", „Sauerkraut" und „SauerkrautSmall". Drei Objekte sind in Abb. 8.10 zu sehen.

Für den Versuch werden sechs Positionen auf dem Tisch vor dem Roboter vorgegeben. An einer der Positionen ist ein Objekt platziert. Der Roboter greift das Objekt skriptgesteuert und legt es an einer anderen Position ab. Dabei werden die gemessenen Sensorwerte abgespeichert und mit der gerade aktiven Manipulationsphase annotiert. Dies wird für jede Position und jedes Objekt wiederholt. Insgesamt werden pro Objekt fünfzehn erfolgreiche Pick-and-Place Zyklen

aufgezeichnet; dabei werden je Objekt zwei bis drei unterschiedliche Griffe angewandt. Die in dieser Phase aufgezeichneten Daten werden als erfolgreich markiert.

In einem zweiten Durchgang werden die selben Transportaktionen ausgeführt. Hier werden jedoch durch den Experimentator künstliche Störungen erzeugt, die den Transport fehlschlagen lassen. Folgende Störungen werden eingesetzt: Entfernen des Objektes vom Tisch, Entfernen des gegriffenen Objektes aus der Hand. Ein dritte Möglichkeit ist das Verschieben des Objektes auf dem Tisch, sodass dieses im Verlauf des Greif- und Transportvorgangs mit hoher Wahrscheinlichkeit aus der Hand fällt. Dies kann jedoch in der noch nicht ausführungsüberwachten Lernphase zu Schäden am Roboter führen, daher wird es hier nur in Situationen verwendet, in denen es als sicher erachtet wird. Konkret bedeutet dies: Für Griffe, deren Hauptanrückrichtung seitlich und nicht von oben orientiert ist. In diesem Durchgang werden für jedes Objekt fünf fehlschlagende Pick-and-Place Operationen durchgeführt. Die aufgezeichneten Daten werden ab dem Eintreten des Fehlers als Fehlschlag markiert.

Die aufgezeichneten Daten werden nun in Trainings- und Testmenge unterteilt. Für jedes Objekt wird ein Erfolgs- und ein Misserfolgsfall als Element der Testmenge ausgewählt, die übrigen Beispiele bilden die Trainingsmenge. Auf den annotierten Daten der Trainingsmenge wird eine SVM trainiert. Die Ergebnisse sind in Tabelle 8.2 dargestellt. Der Trainingsprozess benötigt auf dem beschriebenen Testrechner für die Trainingsmenge 16,9 s.

Zum Vergleich ist dort auch ein Versuch bei dem nur ein Objekt trainiert wurde ausgewertet. In diesem Fall generalisiert das Ergebnis nicht auf die Testmenge. Im Falle der vollständigen, beschriebenen Trainingsmenge klassifiziert die SVM dagegen 99,8 % der Testdaten korrekt. Ein solches Ergebnis legt den Verdacht nahe, dass die Beispiele der so erzeugte Testmenge den Trainingsbeispielen zu ähnlich sind, sodass auch hier keine gute Generalisierung vorliegt. Wendet man die SVM nur auf die Trainigsdaten an, erhält man sogar eine korrekte Klassifikation aller Trainingsbeispiele.

Um dem Verdacht der mangelnden Generalisierung entgegen zu treten, wird eine weitere Testmenge erzeugt. Neben den im ersten Versuch verwendeten Objekten kommen hier zwei weitere Objekte und abgeänderte Objektposen zum Einsatz. Wie bei der Erzeugung der fehlerhaften Beispiele wird in diesen Versuch eingegriffen, damit Objekte nicht genau wie geplant gegriffen werden, sondern z.B. in der Hand verschoben sind.

Als Ergebnis bleibt eine erfolgreiche Klassifikation von 93,4 % der Datenpunkte. Fehlklassifikationen treten dabei als Häufungen innerhalb von einzelnen Aktionen auf, d. h. es wäre mit einem Fehler in 15 Aktionen zu rechnen. Allerdings entstammen die Fehlklassifikationen Greifvorgängen, in denen das Objekt tatsächlich signifikant anders gegriffen wurde, als geplant. Solche Situationen treten in der praktischen Verwendung selten auf und führen auch nicht mehr

Versuch	Abschnitt	Aktionsmodell	Arbeitsraum	Szenen-mechanik	Handlungs-adaption	Ausführungs-überwachung
Kinematik	8.3.1		x		x	
Abhängige Objekte	8.3.2		x	x	x	
Generische Aktionen	8.3.3	x	x		x	
Abweichende Ausführung	8.3.4				x	x
Übergeordnete Steuerung	8.3.5	x		x	x	

Tab. 8.3.: Überblick über die durchgeführten Versuche und beteiligten Komponenten.

Beteiligte Objekte	1
Manipulierte Objekte	1
Geplante Objektaktionen	2
Geplante atomare Aktionen	10
Zeit Planung	2 s
Zeit Physik	-
Zeit bis Ausführung	2 s

Tab. 8.4.: Auswertung der Planung zur Objektübergabe.

sicher zum gewünschten Ergebnis. Die erreichte Klassifikationsgenauigkeit ist damit für die angestrebte Anwendung ausreichend hoch.

8.3. Evaluation des Gesamtsystem an ausgewählten Aufgaben

Nachdem im vorigen Abschnitt die Komponenten des Systems isoliert voneinander evaluiert wurde, wird nun die Funktionalität des Gesamtsystems belegt. Dazu werden zu wichtigen Aspekten der szenenabhängigen Adaption von Handlungssequenzen beispielhaft Szenarien betrachtet, in denen diese Aspekte relevant sind. Es wird untersucht wie das System und seine Komponenten eine Lösung für die Manipulationsaufgabe erzeugen. Einen Überblick über die Versuche und die beteiligten Komponenten wird in Tabelle 8.3 gegeben. Besonders wird zu jedem Experiment die Ausführbarkeit des erzeugten Plans in der Realität und die benötigte Zeit von der Wahrnehmung der Szene bis zum Vorhandensein des vollständigen Plans betrachtet. Im Sinne der Onlinefähigkeit wird dafür eine Wartezeit von weniger als 60 Sekunden als akzeptabel angenommen.

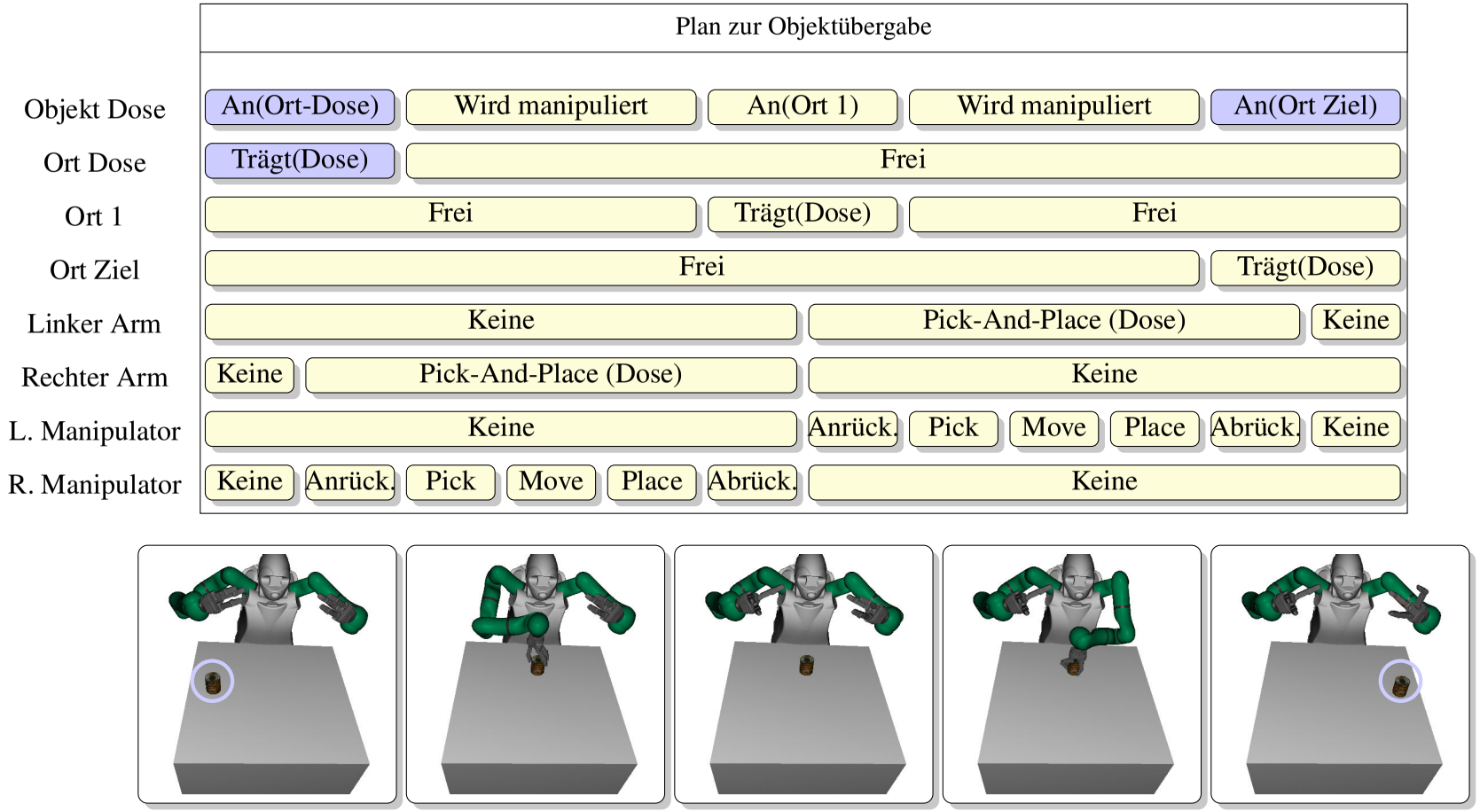
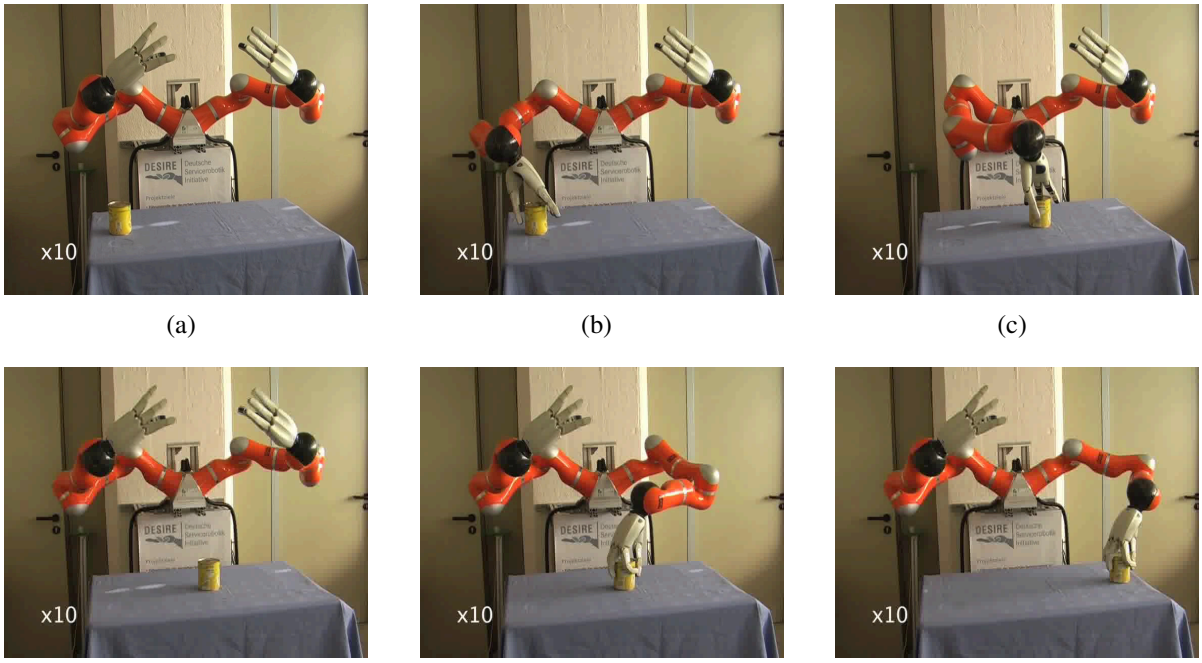


Abb. 8.11.: Gantt Diagramm des an die Roboterkinematik adaptierten Plans für eine Pick-And-Place Aktion. Nur die relevante Teilmenge der Zeitstrahlen ist visualisiert. Gegebene Start- und Zielzustände sind blau markiert. Im unteren Teil des Bildes ist eine simulierte Ausführung des Plans dargestellt. Durch das Einfügen der Zwischenablage in der Mitte des Arbeitsraumes erreicht der Planer ein Ziel, das aufgrund der Kinematik durch eine einzelne Aktion nicht erreichbar ist.



(a) (b) (c)

(d) (e) (f)

Abb. 8.12.: Reale Ausführung der geplanten Handlungssequenz zum Umstellen der Sauerkrautdose durch den Demonstrator. Die Dose wird in der Mitte des Tisches abgestellt, wo sie von beiden Armen gut erreichbar ist. Von dort gelingt der Transport zur Zielposition.

8.3.1. Adaption an die Roboterkinematik

Der zentrale in dieser Arbeit identifizierte Punkt eines Systems zu Handlungsadaption ist die Konsistenz zwischen symbolischem Plan und kontinuierlichem Modell bzw. der echten Welt. In diesem Versuch wird die Fähigkeit eines symbolischen Planers demonstriert, Wissen aus den kontinuierlichen Modellen zu berücksichtigen, das nicht explizit symbolisch modelliert ist. Darüber hinaus wird gezeigt, dass mehrere Aktionen zu einer Sequenz zusammengestellt werden können, um ein Ziel zu erreichen, das durch eine einzelne Aktion aufgrund der Roboterkinematik, also des kontinuierlichen Modells, nicht erreichbar ist.

Es werden die folgenden Komponenten verwendet: Die symbolische Planung erstellt eine Sequenz von Aktionen. Dazu greift sie auf ihre Anbindung an die Manipulationsplaner, in diesem Versuch die inverse Kinematik zurück, um vom Roboter ausführbare Aktionen zu identifizieren. Die Arbeitsraumbeschreibung wird verwendet, um geeignete Ablageorte im gemeinsamen Arbeitsraum beider Manipulatoren zu bestimmen.

In dem Versuch wird auf einem Tisch vor dem Demonstrator eine Konservendose platziert. Diese steht am Rand des Tisches und ist nur von einem Arm, dem (vom Demonstrator aus betrachtet) rechten erreichbar. Die Aufgabe des Systems ist es, die Dose an einem gegebenen

Ort am andern Rand des Tisches abzustellen. Dies ist mit einer einzelnen Pick-and-Place Aktion nicht zu erreichen, da keiner der Arme Anfangs- und Zielort der Dose erreichen kann.

Der erstellte Plan ist in Abb. 8.11 visualisiert. Blau markierte Zustände ergeben sich aus der initialen Szenenkonfiguration bzw. aus dem Zielzustand. Es wurde ein Zeitstrahl für das einzige Objekt der Szene erstellt. Ebenso existieren Orte für die initiale Pose des Objektes und das Ziel. Ein weiterer Ort wurde aufgrund der fusionierten Arbeitsraumbeschreibung des Sauerkrautdosenobjektes in der Mitte des Tisches erzeugt. Er ist in Abb. 8.12(d) gut zu erkennen.

Auf das symbolische Modell mit Start- und Zielzustand wird der symbolische Planer angewandt. Aufgrund der verwendeten iterativen Tiefensuche wird zuerst ein Plan mit nur einer Aktion erstellt. Dieser soll die Dose direkt vom Start zum Ziel bewegen. Bei der Auswahl eines geeigneten Manipulators sind die Anfragen an die Manipulationsplanung negativ, keiner der Arme kann die Aktion im Kontinuierlichen ausführen. Die Planung schlägt fehl. Darauf erhöht die iterative Tiefensuche die maximale Planlänge auf zwei Objektaktionen. Wieder schlägt der direkte Transport fehl, nun kann jedoch ein Transport zu einem andern Ort erzeugt werden, der im Laufe der Planung zu Ort 1 festgelegt wird. Von diesem kann dann die Aktion zum Zielort erzeugt werden. Die Auswahl der Manipulatoren lässt für jeden Transport nur den Manipulator zu, der die Bewegung kinematisch, also im kontinuierlichen Modell, ausführen kann.

Zur Erstellung des Plans benötigte eine Intel(R) Core i7 CPU 920 mit 2.67 GHz und 3 Gb RAM knapp 2 s. Dies schließt die symbolische Planung sowie die Manipulationsplanung mit ein. Die Zeit ist nur unter Vorbehalt mit den in den folgenden Versuchen gemessenen Zeiten vergleichbar, da der verwendeten Planerimplementierung die hierarchisches Zerlegung des symbolischen Modells fehlt. Zu erwarten ist, dass die Zeiten mit der vollständigen Implementierung etwas schneller sind. Unabhängig davon ist auch die gemessene Zeit für die Online-Anwendung des Verfahrens schnell genug.

Der erzeugte Plan wurde in der Simulation und auf dem realen Demonstrator erfolgreich ausgeführt. Die reale Ausführung ist in Abb. 8.12 dargestellt. In der Ausführung ist die erzeugte Objektpose für beide Arme zu erreichen, das Greifen des Objektes ist erfolgreich. Die Arbeitsraumbeschreibung und die Erzeugung von Zwischenablageposen ist für diese Anwendung geeignet. Ebenso werden durch den Planer nur kinematisch ausführbare Aktionen erzeugt, obwohl das Wissen darüber nicht im symbolischen Modell enthalten ist. Auch die Integration von kontinuierlichen und symbolischen Modellen bzw. Planern entspricht also den Anforderungen. Zuletzt ist die durch das System erzeugte Handlungssequenz geeignet, das gegebene Ziel zu erreichen. Der Versuch wurden mit weiteren kinematisch erreichbaren Start und Zielposen mehrfach erfolgreich wiederholt.

Beteiligte Objekte	4
Manipulierte Objekte	2
Geplante Objektaktionen	2
Geplante atomare Aktionen	10
Zeit Planung	15 s
Zeit Physik	29 s
Zeit bis Ausführung	49 s

Tab. 8.5.: Auswertung der Planung mit abhängigen Objekten.

8.3.2. Adaption aufgrund abhängiger Objekte

Das System zur Handlungsadaption muss in der Lage sein, Abhängigkeiten zwischen den Objekten zu berücksichtigen. Dazu wurden in dieser Arbeit die mechanischen Relationen vorgestellt. Die Komponente zu ihrer Erzeugung wurde bereits in Abschnitt 8.2.2 für sich alleine betrachtet. In diesem Versuch wird untersucht, ob die erzeugten mechanischen Relationen im Planungssystem integriert geeignet sind, mechanische Abhängigkeiten zu berücksichtigen.

Drei Komponenten spielen dabei eine Rolle. Die Erzeugung mechanischer Relationen wurde als zentrale Komponente bereits erwähnt. Um die durch sie erkannten Abhängigkeiten auflösen zu können, werden Orte zum Ablegen von Objekten benötigt. Sie entstammen der Arbeitsraumbeschreibung. Wie in allen Versuchen im Gesamtsystem kommt der symbolische Planer zum Einsatz, um aufgrund der erzeugten symbolischen Beschreibung zielführende Handlungssequenzen zu erzeugen.

In diesem Versuch steht ein Stapel aus vier Objekten auf einem ansonsten freien Tisch vor dem Roboter. Die Szene ist in Abb. 8.14(a) dargestellt, eine Abbildung des Stapel findet sich ebenfalls in Abb. 6.13(a). Die Aufgabe ist es, die Ceylonteeschachtel am (vom Roboter aus betrachtet) linken Rand des Tisches abzustellen (Abb. 8.14(h)). Intuitiv ist klar, das dazu die Dose, die auf dem Teeobjekt steht, umgestellt werden muss.

Aufgrund der Objektposen aus der Wahrnehmung werden Ortsymbole für alle Objekte erzeugt. Ortsymbole, die nicht auf dem Tisch liegen, sondern auf anderen Objekten werden dabei markiert, sodass sie nicht als Ablageort in Frage kommen. In Abb. 8.13 ist aus Platzgründen lediglich der Zielort der Manipulation visualisiert. Aufgrund der Arbeitsraumbeschreibung werden weitere Orte, die auf den Tisch verteilt sind, erzeugt. Dabei werden die Greifbarkeitskarten der Arme für ein Objekt über die Maximum-Funktion fusioniert. Eine aufgabenspezifische Fusion findet nicht statt, da lediglich erreichbare Orte gesucht sind. Für jedes Objekt werden zwei erreichbare Orte erzeugt. Die erzeugten Orte werden zusammengefasst, sodass ein Mindestabstand von 20 cm zwischen ihnen besteht. Auch von bereits existierenden Orten wird dieser Abstand eingehalten. Jedes Objekt kann auf jedem erstellten Ort platziert werden, auch wenn

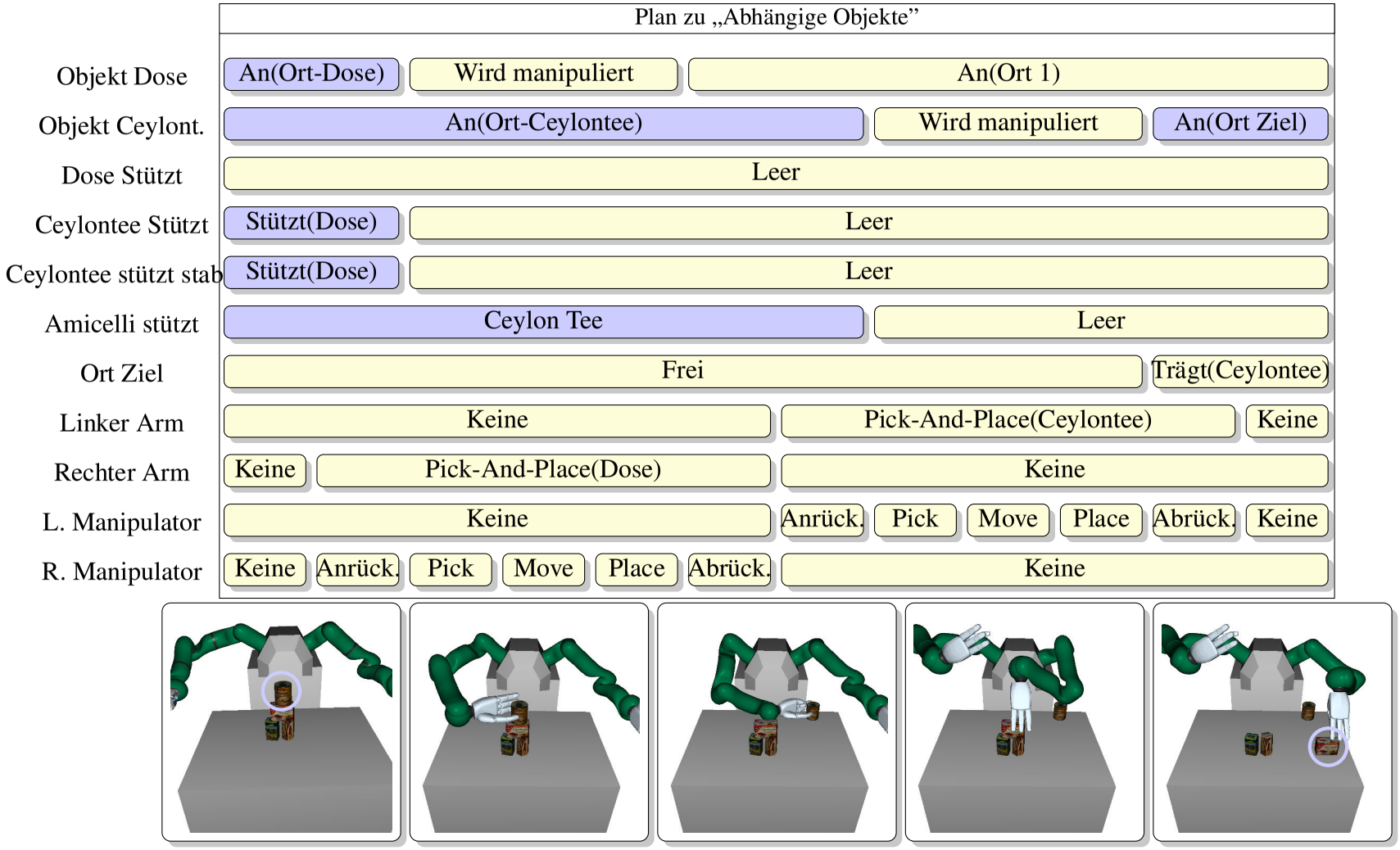
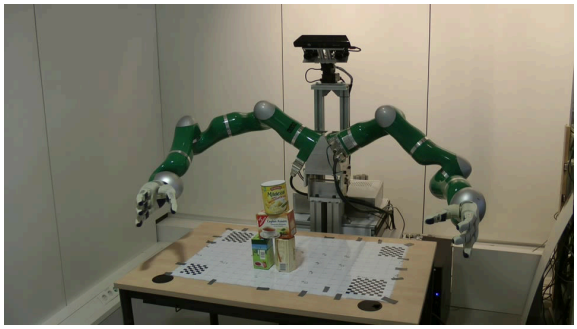
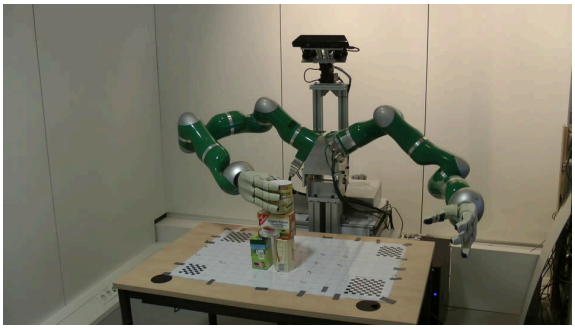


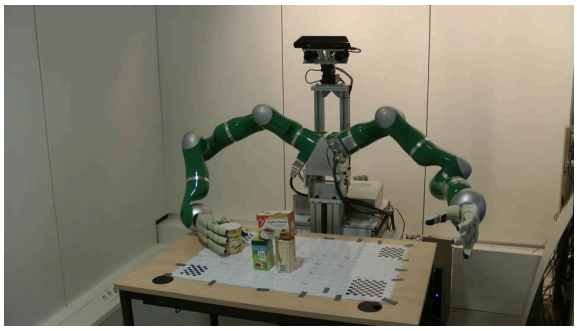
Abb. 8.13.: Gantt Diagramm des Plans für eine Pick-And-Place Aktion. Nur die relevante Teilmenge der Zeitstrahlen ist visualisiert. Gegebene Start- und Zielzustände sind blau markiert. Im unteren Teil des Bildes ist eine simulierte Ausführung des Plans dargestellt.



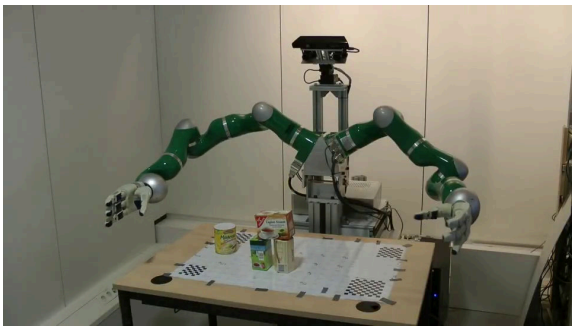
(a)



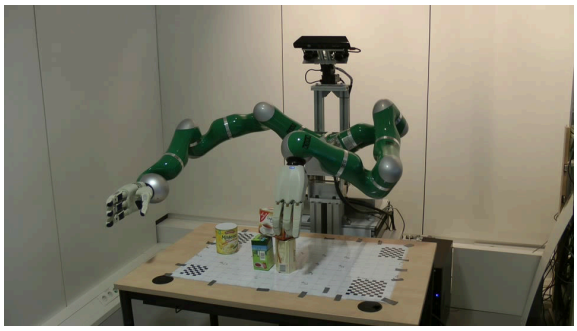
(b)



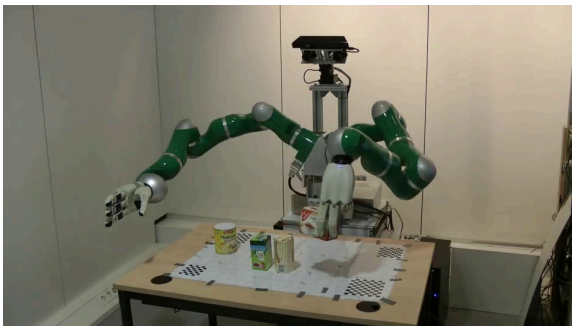
(c)



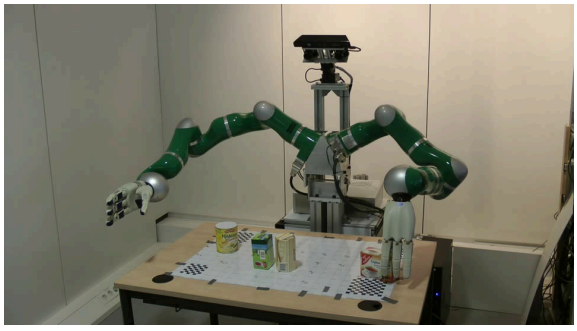
(d)



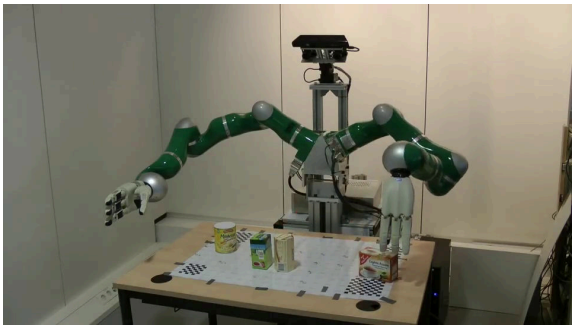
(e)



(f)



(g)



(h)

Abb. 8.14.: Erfolgreiche Ausführung des Plans zum Ablegen der Ceylonteeschachtel. Aus den mechanischen Relationen schließt das System, dass die Teeschachtel nicht bewegt werden kann, solange diese die Dose stützt. Daher wird zuerst die Dose auf einem aufgrund der Arbeitsraumbeschreibung erzeugten Ort abgestellt. Im Anschluss kann der Roboter die Teeschachtel an ihrer Zielposition platzieren.

Beteiligte Objekte	6
Manipulierte Objekte	3
Geplante Objektaktionen	3
Geplante atomare Aktionen	11
Zeit Planung	12 s
Zeit Physik	-
Zeit bis Ausführung	12 s

Tab. 8.6.: Auswertung der Planung mit generischen Aktionen

der Ort nicht aufgrund der Greifbarkeit des Objektes erzeugt wurde. Die Anbindung an die Manipulationsplaner sichert die erfolgreiche Ausführung.

Aufgrund der Objekte und ihrer Positionen erzeugt die Symbolisierung die mechanischen Relationen. Der resultierende Szenengraph ist in Abb. 6.13(c) zu sehen. Die Abbildungen auf Zeitstrahlen sind in Abb. 8.13 blau markiert.

Bei der Planung tritt zu Beginn ein Konflikt zwischen dem Startzustand „Cylontee An(Ort-Ceylontee)" und dem Zielzustand „Cylontee An(Ort-Ziel)" auf. Das wird vom Planer durch Einfügen von „Wird Manipuliert" auf dem Zeitstrahl „Objekt Cylontee" aufgelöst. Dieser erfordert wiederum, dass „Ceylontee Leer" gilt. Nachdem der Zeitstrahl wie in Abb. 8.13 dargestellt geplant ist, erzwingt der Übergang von „Stützt (Dose)" zu „Leer" auf „Ceylontee" den „Wird manipuliert" Zustand von „Objekt Dose", der mit dem Ende des „Stützt (Dose)" Zustands beginnt. Damit ist der Plan im Wesentlichen festgelegt, die weiteren Zeitstrahlen können konsistent gefüllt werden.

Auf einem PC mit Intel Core2 Duo T7800 CPU, 2.60 GHz und 4 GB RAM benötigt die Planung der Aktionssequenz 15 s, wovon 8 s von für Anfragen an die Manipulationsplaner zur Berechnung der inversen Kinematik benötigt werden, 7 s werden zum Durchsuchen des Zustandsraums verwendet. Weitere 29 s sind für die Berechnung der Szenenrelationen nötig. Mit Initialisierung der Planer werden 49 s von der Anfrage bis zum vollständigen Plan benötigt. Die Zeiten sind in Tabelle 8.5 zusammengefasst. Sie befinden sich innerhalb der angestrebten maximalen Planungszeit.

Der erzeugte Plan wurde auf einem bimanuellen Robotersystem erfolgreich ausgeführt. Das Ergebnis ist in Abb. 8.14 abgebildet. Alle geplanten Aktionen sind ausführbar und erhalten den geordneten Zustand der Szene; d. h. es fallen keine Objekte und es kommt zu keinen Kollisionen. Die erzeugten Szenenrelationen sind also in der Lage, die Abhängigkeiten zwischen den Objekten korrekt und für den symbolischen Planer nutzbar zu modellieren.

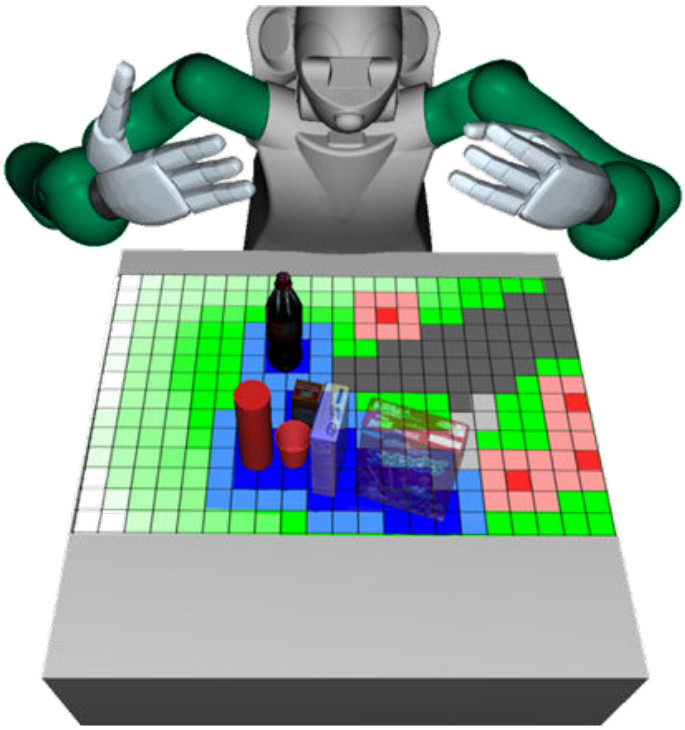


Abb. 8.15.: Visualisierung der Ortssymbole im Einschenkversuch. Blaue Orte ergeben sich aus den Positionen der Objekte in der initialen Szene. Im Hintergrund in Grüntönen dargestellt ist die Greifbarkeit der Sauerkrautdose für den linken Arm. Rot sind daraus entstandene Ablageorte, die für den linken Arm erreichbar sind. In blasseren Blau- und Rottönen dargestellt sind Orte, die aufgrund von Abstandsregeln zu bestehenden Orten nicht verwendet werden können. Graue Orte müssen beim Ausführen des Einschenkens frei sein, um Kollisionen zu vermeiden.

8.3.3. Berücksichtigung generischer Aktionen

Manipulation ist nicht auf Pick-and-Place Sequenzen beschränkt; eine Vielzahl anderer Aktionen ist denkbar. Solche Aktionen können von unabhängigen Manipulationsplanern erzeugt werden. In diesem Experiment wird am Beispiel einer Einschenkaktion, die dem Planer von (Jäkel, 2013) entstammt, gezeigt, dass das vorgestellte System in der Lage ist, mit generischen Aktionen aus Manipulationsplanern umzugehen.

Neben den bereits an den vorigen Versuchen beteiligten Komponenten zur symbolischen Planung und Arbeitsraumsymbolisierung wird in diesem Versuch die Komponente zur Aktionssymbolisierung durch Freiraumanforderungen besonders betrachtet.

Die für den Versuch gestellte Szene beinhaltet einen Becher im Mittelbereich des Tisches. In ihn soll aus der (offenen) Flasche auf dem mittleren hinteren Bereich des Tisches eingeschenkt werden. Der Manipulationplaner berücksichtigt dabei ebenso das Greifen der Flasche wie auch das eigentliche Einschenken. Für beide Teile der Aktion wird Freiraum benötigt. Beim Einschenken muss die Öffnung der Flasche so über dem Becher gehalten werden, dass die Flüs-

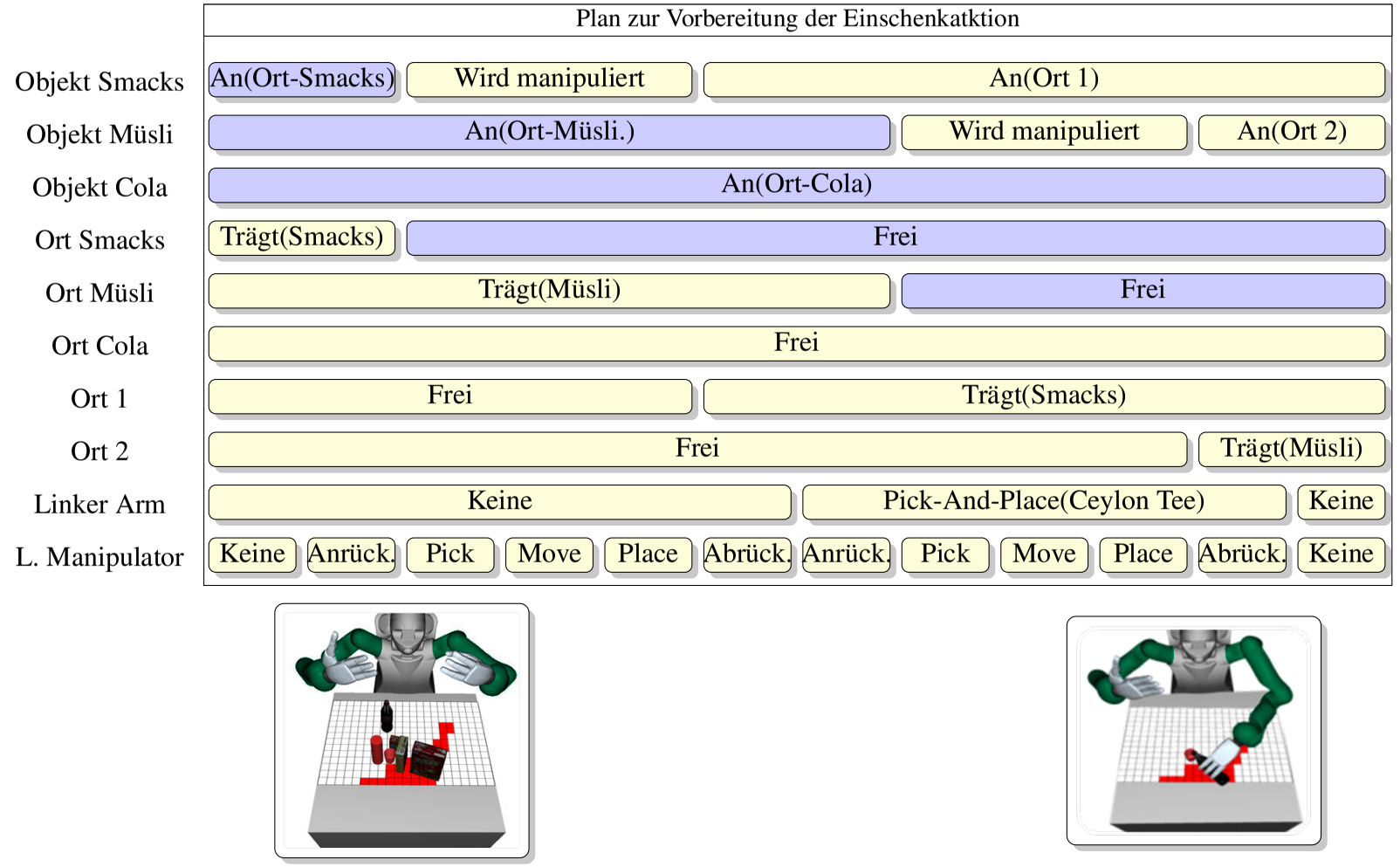
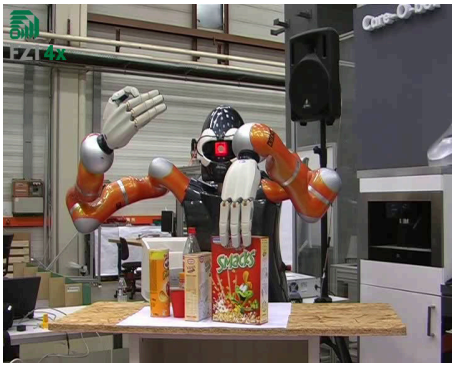
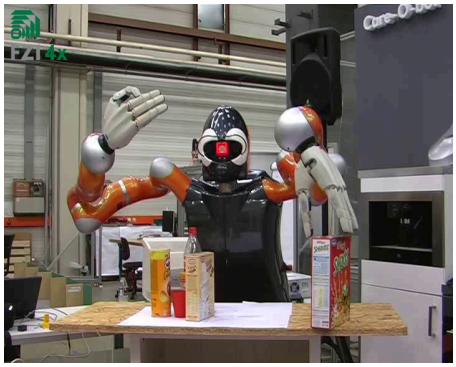


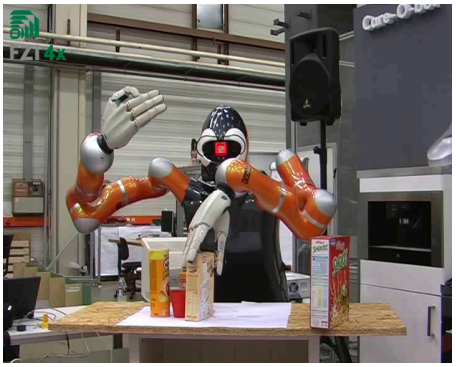
Abb. 8.16.: Gantt Diagramm des Plans zur Vorbereitung einer Einschenkaktion. Die Einschenken Aktion selbst ist nicht visualisiert, lediglich die geplanten Aktionen, die ihre Ausführung ermöglichen. Nur die relevante Teilmenge der Zeitstrahlen ist visualisiert. Blau markierte Startzustände entstammen der unten links visualisierten initalen Szene. Die blau markierten Zielzustände werden aus den Freiraumanforderungen der Einschenkaktion erzeugt, wie sie unten rechts dargestellt sind.



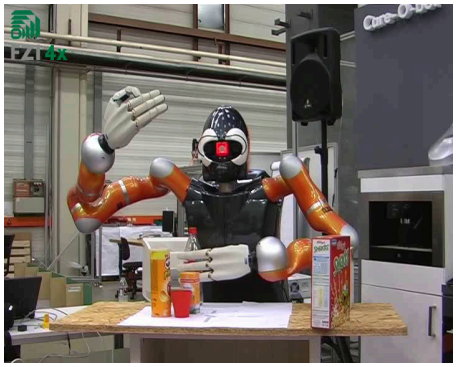
(a)



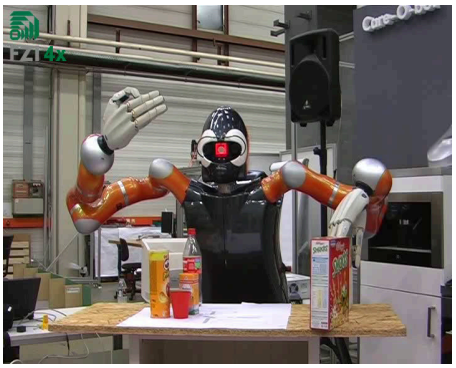
(b)



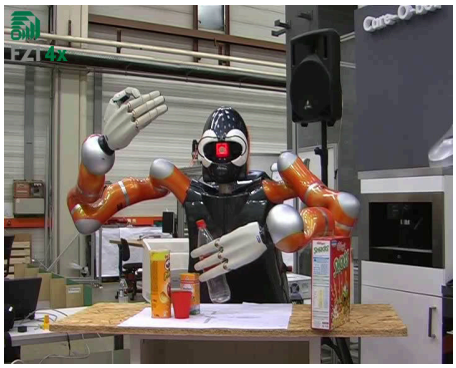
(c)



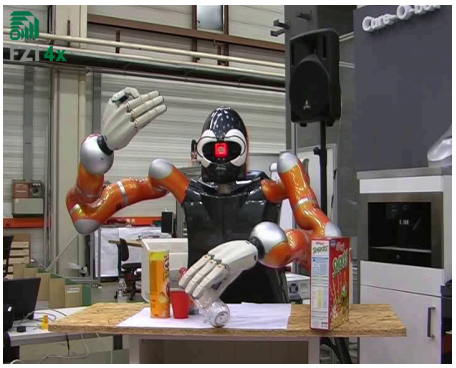
(d)



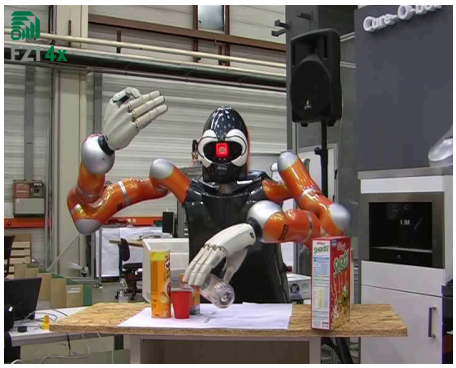
(e)



(f)



(g)



(h)

Abb. 8.17.: Erfolgreiche reale Ausführung der Einschenksequenz. Um den Einschenkvorgang auszuführen benötigt der Roboter genügend Freiraum neben dem Glas. Um diesen zu schaffen werden Aktionen geplant und ausgeführt, welche die beiden Müslischachteln an den Tischrand bewegen.

sigkeit in diesen fließen kann. Im Versuchsaufbau sind vier Hindernisobjekte um den Becher angeordnet: eine zylinderförmige Chipsschachtel, sowie die quaderförmigen Salz-, Müsli- und Smacksschachteln. Solange alle diese Objekte an ihrer initialen Position stehen, ist es nicht möglich, die Einschenkaktion auszuführen. Das System ist gezwungen, die Szene umzustellen.

Aus der in Abb. 8.17(a) abgebildeten Szene werden die Symbole zur Beschreibung der Objektpositionen erzeugt. Ebenso werden Ablageorte aufgrund der Arbeitsraumbeschreibung erzeugt. Die Behandlung der unterschiedlichen Objekte erfolgt wie in Abschnitt 8.3.2 beschrieben. Mechanische Relationen werden in diesem Versuch nicht erstellt. Zum Erstellen der Freiraumbedingungen wird der Manipulationsplaner für das Einschenken auf eine leere Szene, die lediglich den Becher und die Cola Flasche enthält, angewandt. Eine resultierende Konfiguration ist in Abb. 8.16 unten rechts visualisiert. Die Armkonfiguration ist rot auf den Tisch projiziert, Objekte auf diesen Orten würden zu Kollisionen führen, falls sie höher wären als der vertikale Abstand des Arms zum Tisch. Die geplante Trajektorie wird abgetastet und für jede Konfiguration die Projektionen auf den Tisch zur Vereinigungsmenge fusioniert. Sie beschreibt den benötigten Freiraum der Aktion.

Alle erzeugten Ortssymbole des Versuchs und ihre Abhängigkeiten sind in Abb. 8.15 visualisiert. Die Freiräume, die für die Einschenkaktion benötigt werden, sind grau hinterlegt, sofern sie nicht von Objekten belegt sind. Orte, die im initialen Zustand Objekte beinhalten, sind blau dargestellt; hellblau ist ein durch das Objekt blockierter Bereich, der nicht zum Abstellen verwendet werden kann. Die Grefbarkeit des Smacksschachtelobjektes ist in Grüntönen visualisiert. Vier rote Orte wurden zum Abstellen von Objekten erzeugt und können vom linken Arm erreicht werden. Auch hier ist eine hellrote Umgebung markiert, in der keine weiteren Orte aus der Arbeitsraumbeschreibung erzeugt werden.

Der erstellte Plan ist in Abb. 8.16 als Gantt Diagramm visualisiert. Die „Frei”-Symbole auf den Zeitstrahlen des „Ort Smacks” und „Ort Müsli” aus den Vorbedingungen der Einschenkaktion erzeugen Konflikte mit den „Trägt” Symbolen der initialen Konfiguration auf ebendiesen Zeitstrahlen. Der symbolische Planer kann diese durch Einfügen von „Wird Manipuliert” Zuständen und konsistentes Ausfüllen der Zeitstrahlen, insbesondere mit Pick-and-Place Aktionen, beheben. Am Ende des visualisierten Plans ist die Szene in einem Zustand, in der das Einschenken möglich ist. Die Einschenkaktion selbst ist in Abb. 8.16 aus Platzgründen nicht visualisiert; sie schließt sich direkt an und wird durch die Hierarchieebenen unverändert nach unten propagiert.

Auf einem PC mit Intel Core2 Duo T7800 CPU, 2.60 GHz und 4 GB RAM benötigt die Planung der Aktionssequenz 12 s. Nicht berücksichtigt ist dabei der Manipulationsplaner, der im Versuch offline verwendet wurde. Mechanische Szenenrelationen wurden nicht verwendet, sodass nach 12 s ein ausführbarer Plan vorhanden ist. Die Zeiten sind in Tabelle 8.6 zusammengefasst.

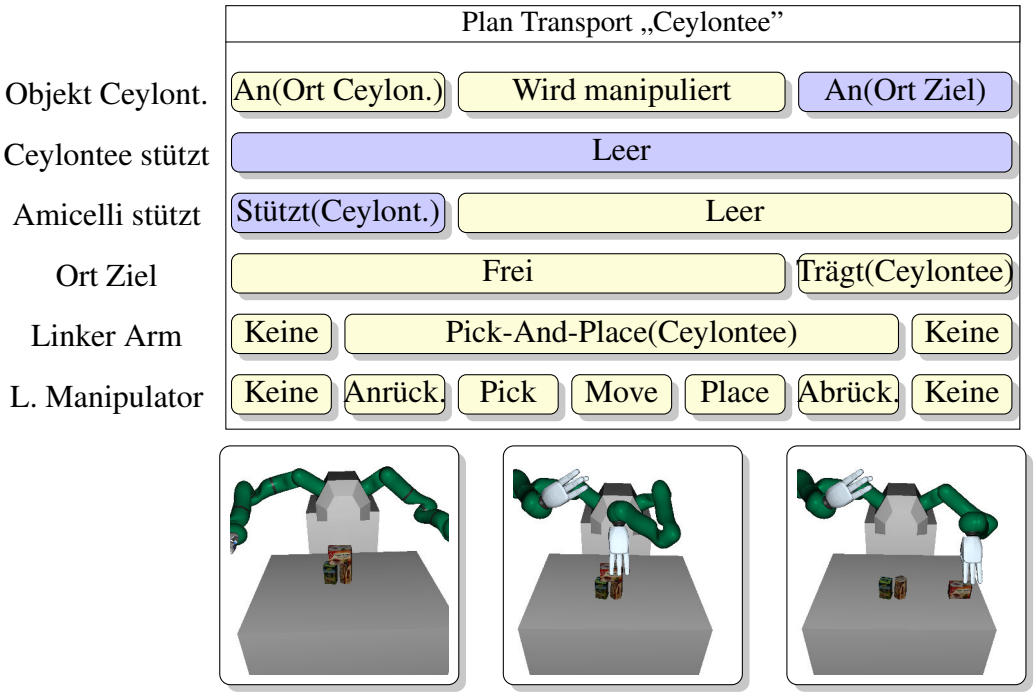


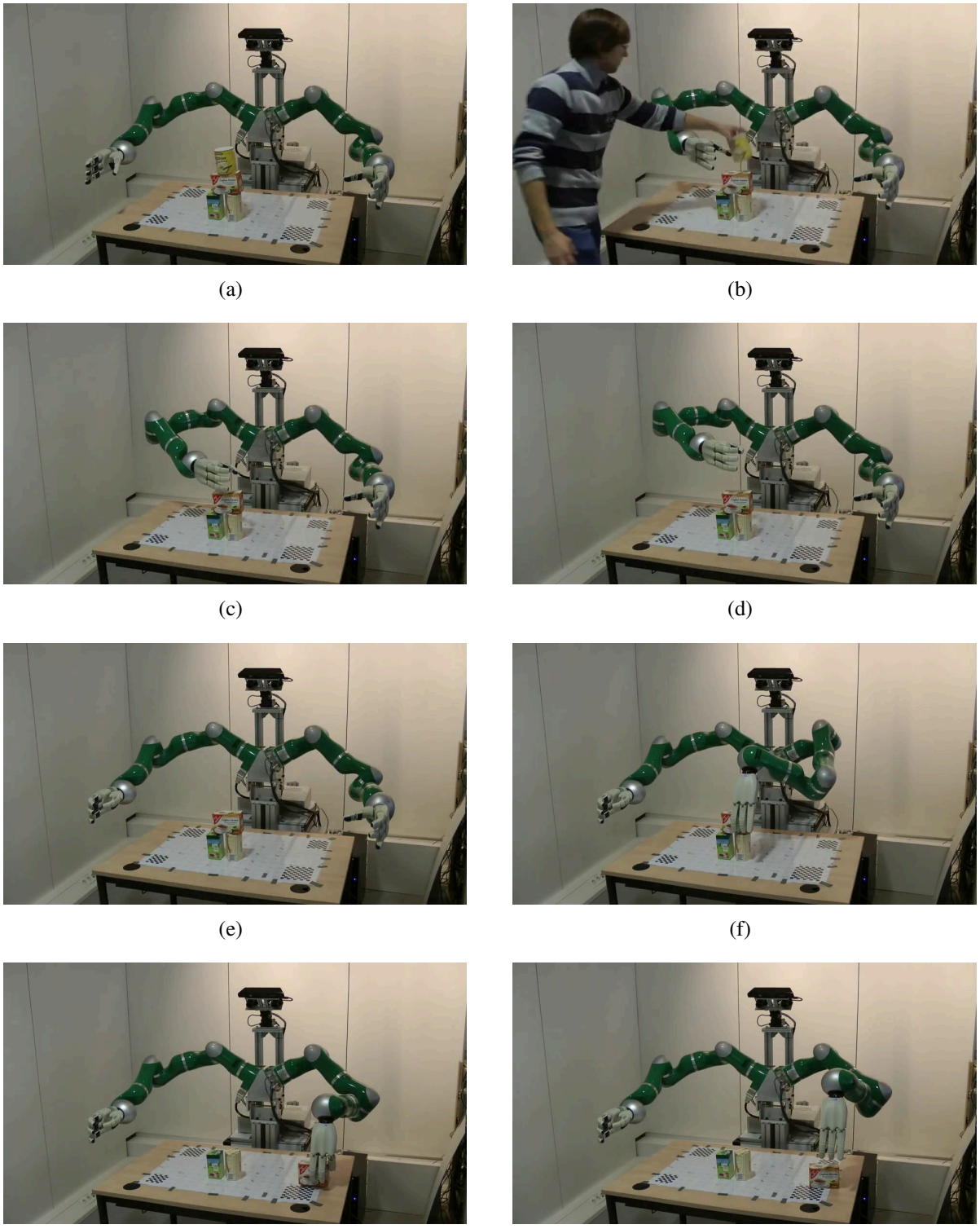
Abb. 8.18.: Gantt Diagramm des neuen Plans, nachdem die Dose aus der Szene entfernt wurde. Das unveränderte Ziel kann mit einer einzigen Pick-And-Place Aktion erreicht werden.

In Abb. 8.17 ist die reale Ausführung zu sehen. Die Umräumaktionen bestätigen die Ergebnisse der vorangegangenen Experimente. Das System ist in der Lage real ausführbare, mit der kontinuierlichen Roboterkinematik konsistente Pläne zur Anpassung einer Szene zu erstellen. Ab Abb. 8.17(f) beginnt die eigentliche Einschenkaktion. Der Manipulationsplaner ist in der erstellten Szene in der Lage, diese ausführbar zu planen. Auch hier erfüllt das System die Anforderungen.

Der zur Planung der Einschenkaktion verwendet Planer kann unterschiedliche Trajektorien erstellen, die das Einschenken umsetzen. In einem weiteren, nicht abgebildeten, Versuch wurde eine Trajektorie für den linken Arm erzeugt, die sich eher im rechten Teil des Arbeitsraumes befindet. Wird diese als Grundlage für die Freiraumanforderungen verwendet, so ist lediglich das Chipsschatelobjekt ein Hindernis. In diesem Fall erstellt das System einen Plan, der das Objekt entfernt. Auf der resultierenden Szene kann das Einschenken geplant und ebenso erfolgreich ausgeführt werden.

8.3.4. Adaption an abweichende Ausführung

Die Ausführung von Manipulationsaktionen in der realen Welt kann anders verlaufen, als vom Planer vorhergesehen; ein Serviceroboter muss in der Lage sein, auf veränderte Abläufe zu reagieren. Die Komponente zur Ausführungsüberwachung wurde in Abschnitt 8.2.3 bereits iso-



(a) (b) (c) (d) (e) (f) (g) (h)

Abb. 8.19.: Reale Ausführung einer Manipulationssequentz mit Störung. Das Objekt, das in der ersten Aktion gegriffen werden soll, wurde vom Experimentator entfernt. Das System erkennt die fehlgeschlagene Greifaktion und löst die Neuplanung der Sequenz aus. Auf der neu wahrgenommenen Szene kann die Manipulation des Ceylontee Objektes ausgeführt werden.

Beteiligte Objekte	5
Manipulierte Objekte	3
Geplante Objektaktionen	3
Geplante atomare Aktionen	11
Zeit Planung	5 s
Zeit Physik	6 s
Zeit bis Ausführung	12 s

Tab. 8.7.: Auswertung der Planung mit Missionssteuerung

liert betrachtet. In diesem Experiment wird ihr Zusammenspiel mit dem symbolischen Planer untersucht. Als dritte Komponente ist die Symbolisierung der mechanischen Relationen an dem Experiment beteiligt.

Der Versuchsaufbau basiert auf der selben Objektanordnung wie in Abschnitt 8.3.2 und ist in Abb. 8.19(a) zu sehen. Entsprechend erzeugt das System auch den in Abb. 8.13 visualisierten Plan. Der Ablauf unterscheidet sich vom ursprünglichen Versuch dadurch, dass während der Ausführung das Dosenobjekt, kurz bevor es gegriffen wird, vom Experimentator entfernt wird (Abb. 8.19(b)).

Anstelle des Dosenobjektes greift der Roboter ins Leere (Abb. 8.19(c), Abb. 8.19(d)). Die Ausführungsüberwachung erkennt den Fehler und bricht die Ausführung des Plans ab. Ausgehend vom Zustand in Abb. 8.19(e) wird die Szene neu wahrgenommen und auf dem erkannten Zustand neu geplant. Der entstandene Plan ist in Abb. 8.18 dargestellt. In der Konfiguration kann der Zielzustand durch nur eine einzige Pick-and-Place Aktion erreicht werden. Ihre erfolgreiche Ausführung ist in Abb. 8.19(f) bis Abb. 8.19(h) zu sehen.

Die Ausführungszeiten wurden in diesem Versuch nicht im Detail bestimmt. Insgesamt benötigt das System 20 s vom Erkennen des Fehlers bis zum Beginn der Ausführung des neuen Plans. Zwei zusätzliche Aktionen kommen durch die Neuplanung hinzu: Beim Erkennen des Fehlers wird der rechte Arm in seine Ruheposition gefahren, zu Beginn des neuen Plans fährt die Hand wieder an die Stelle, an die der Plan abgebrochen wurde. Dies ist nötig, da der Arm sich sonst im Sichtfeld der Sensorik befände.

8.3.5. Handlungsadaption im Zusammenspiel mit einer übergeordneten Missionssteuerung

Der vorgestellte Ansatz zur Planung für einen Serviceroboter setzt ein umfangreiches Wissen über die Umgebung voraus. So wird angenommen, dass Objektpositionen bekannt sind, bzw. durch die Sensorik des Roboters wahrgenommen werden können. Dies ist in einer Haushaltsumgebung nicht immer der Fall. Objekte können sich z.B. in anderen Räumen befinden. Eine

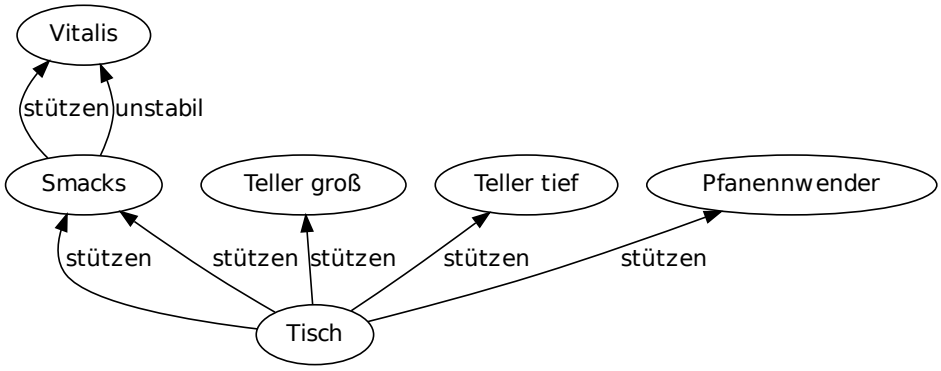
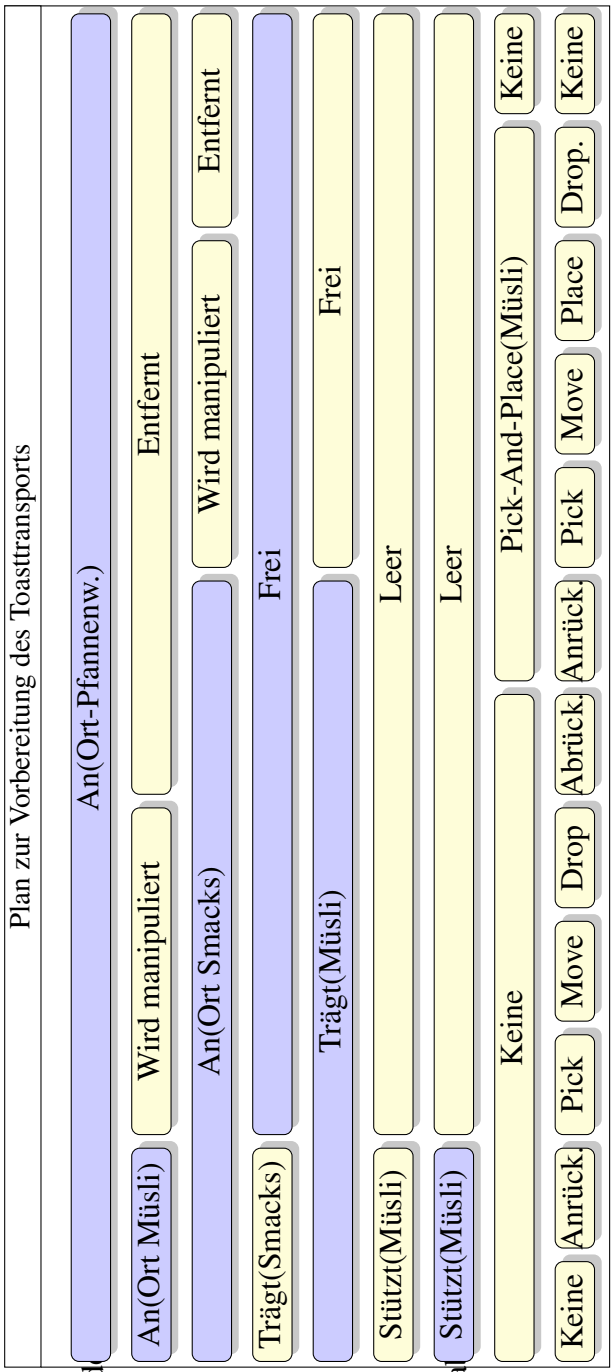


Abb. 8.20.: Szenengraph der Pfannenwender Szene

Suche erfordert Mobilität und geeignete Suchstrategien. Solche Missionen zu Planen geht über die Anforderungen der Handlungssequenzen zur Manipulation hinaus. Ebenso wie bei der Berücksichtigung der von der Abstraktion her niedriger angesiedelten Manipulationsplaner muss das Problem durch die Kooperation mit weiteren, spezialisierten Komponenten angegangen werden. In diesem Experiment wird daher die Integration der Handlungsadaption in eine übergeordnete Missionssteuerung betrachtet.

Es sind drei Komponenten des Systems eingebunden. Die Aktionsmodellierung beschreibt die benötigten Freiräume für die Manipulation mit dem Pfannenwender. Durch die Szenenmechanik wird die Abhängigkeit der Objekte bestimmt und die symbolische Planung fügt Aktionen zu einer zielführenden Sequenz zusammen. Neben den Komponenten, die dieser Arbeit entstammen, kommt der bereits beim „Einschenken" Versuch in Abschnitt 8.3.3 eingebundene Manipualationsplaner aus (Jäkel, 2013) zur Trajektorienplanung zur Handhabung des Pfannenwenders zum Einsatz. Als Missionssteuerung wird das POMDP basierte System aus (Schmidt-Rohr, 2013) angebunden.

Entsprechend des erweiterten Rahmens des Versuches beschränkt sich der Aufbau in diesem Versuch nicht auf einen Tisch. Es wurde ein Raum mit Regalen und einem Tisch zur Manipulation hergerichtet. Dazwischen befindet sich Freiraum, in dem der Roboter mit Hilfe seiner mobilem Plattform fahren kann. Vor dem Roboter befindet sich in Sichtweite, jedoch nicht in direkter Reichweite seiner Arme, der Tisch zur Manipulation (Abb. 8.22(a)). Auf dem Tisch befinden sich ein Pfannenwender, der als Werkzeug zur Manipulation eingesetzt wird, zwei Schachteln, die als Hindernisse platziert werden sowie ein großer Teller. Zwei weitere Becher an den Ecken des Tisches werden als Landmarken für die Lokalisierung eingesetzt und spielen für die Manipulation keine Rollen. Ein weiterer kleiner Teller befindet sich im Regal hinter dem Roboter, initial für diesen unsichtbar. Für die Storyline befindet sich eine Scheibe Toast auf dem



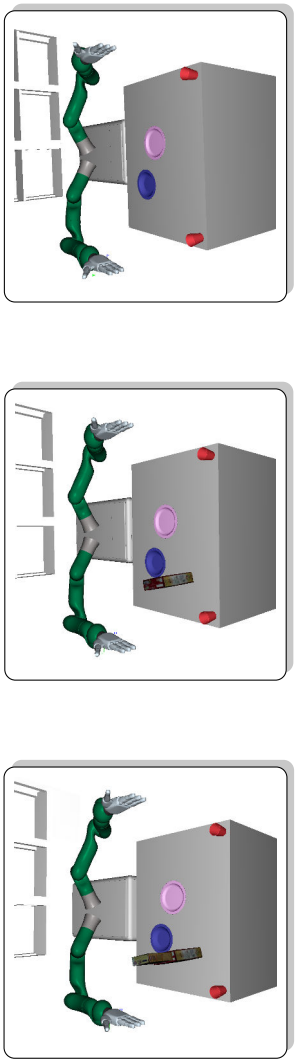
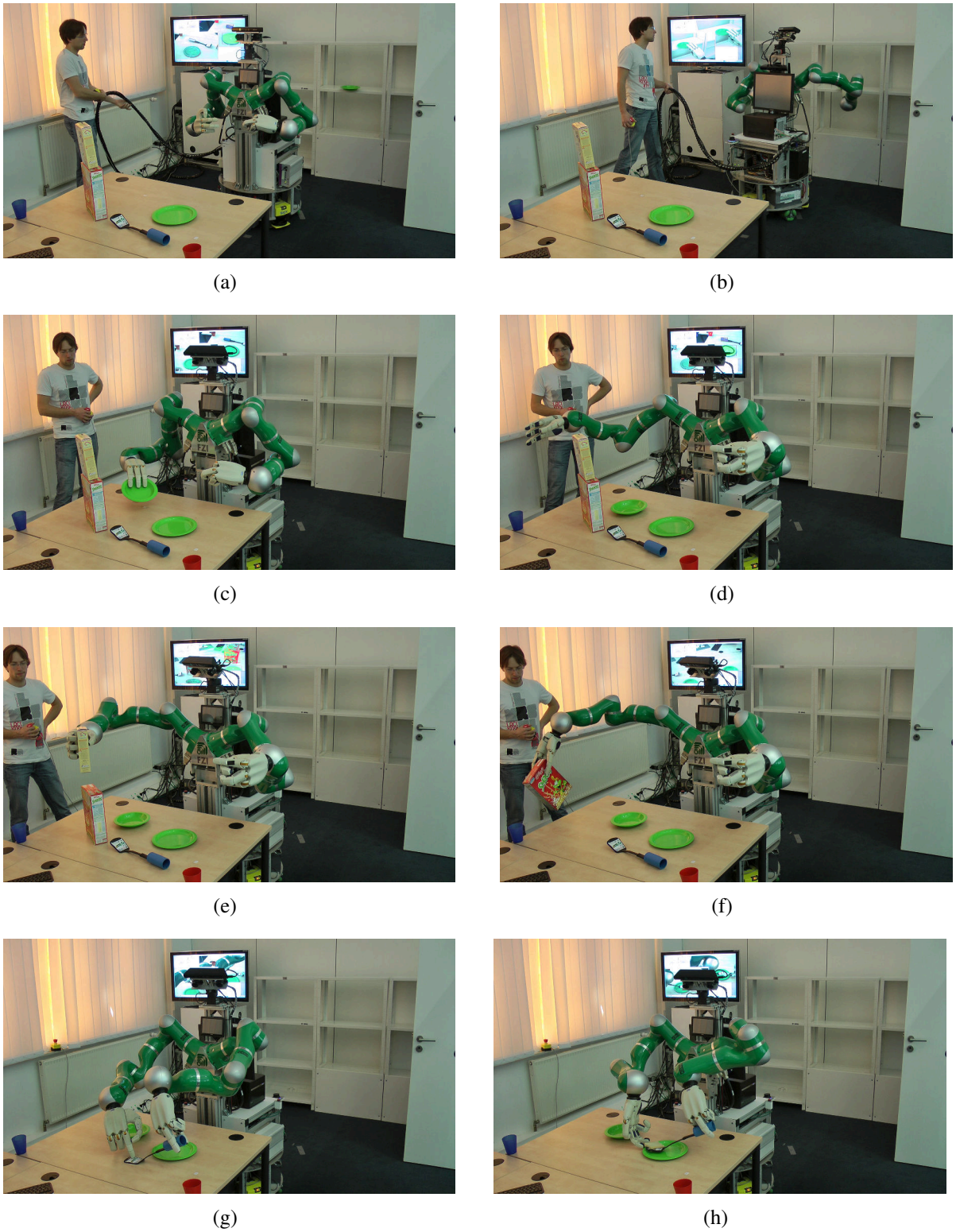


Abb. 8.21.: Gantt Diagramm des Plans zur Vorbereitung der Pfannenwendermanipulation.



(a) (b) (c) (d) (e) (f) (g) (h)

Abb. 8.22.: Reale Ausführung einer Manipulationssequentz mit höherer Steuerung. Als Mission soll eine Scheibe Toast von einem Teller auf einen anderen transportiert werden. Dazu wird ein Pfannenwender eingesetzt. Zuerst muss der Roboter jedoch den zweiten Teller aus dem Regal im Hintergrund holen. Zum Umgang mit unbekannten Objekten wird die höhere Steurungsebene benötigt, welche die abstrakten Aktionen zum Holen des Tellers erstellt. Das Greifen des Pfannenwenders benötigt eine Anpassung der Szene, bei der das vorgestellte System die Aktionen plant. Die Manipulation des Toasts selbst wird durch einen externen Manipulationsplaner geplant.

großen Teller. Da die verwendete Objeklokalisierung den Teller mit Toast nur unzureichend erkennt, wird der Toast erst im Laufe der Ausführung durch den Experimentator in der Szene platziert.

Als Aufgabe soll der Roboter den Toast von einem Teller auf den anderen legen. Die erste Teilaufgabe betrifft dabei die Missionssteuerung: Sie stellt fest, dass der zweite Teller fehlt und erstellt eine Aktionsfolge, diesen Teller im Regal zu suchen und zum Tisch zu transportieren. Die Abbildung der abstrakten Befehle auf den Roboter erledigt das in dieser Arbeit präsentierte System. Die erstellten Pläne im ersten Teil bestehen nur aus einzelnen Aktionen und liefern keine neuen Erkenntnisse; sie werden hier daher nicht weiter behandelt. Die Missionssteuerung ist nicht Teil dieser Arbeit.

Im zweiten Teil des Versuchs muss der Roboter den Pfannenwender greifen. Dies ist nicht ohne weiteres möglich, da der Griff auf dem Tisch liegt und sein Greifen zu einer Kollision des Greifers mit dem Tisch führt. Um dies zu verhindern, wird durch den Manipulationsplaner aus (Jäkel, 2013) eine bimanuelle Trajektorie erzeugt, die erst mit einer Hand die Auflagefläche des Pfannenwenders in Richtung des Tisches drückt und damit den Griff anhebt (Abb. 8.22(g)). So kann die andere Hand den Pfannewender greifen. Die Ausführung dieser Aktion wird in der aufgestellten Szene durch die beiden Schachteln verhindert. Da die Ausführung der Pfannenwendermanipulation durch das hier vorgestellte System gesteuert wird, kann hier die Handlungsadaption greifen und die Szene für die Manipulation verändern.

Die Wahrnehmung der Szene basiert auf den Informationen des Stereokamerasystems im Roboterkopf. Aufgrund der angeschlossenen Objektlokalisierung werden die Objektpositionen in das kontinuierliche Modell und darauf basierend in das symbolische Modell eingetragen. Basierend auf der Szene werden die mechanischen Relationen wie in Abb. 8.20 dargestellt erzeugt. Die Abhängigkeit zwischen Smacks- und Vitalisobjekt ist korrekt dargestellt. Aus dem Manipulationsplaner wird das Aktionsmodell für die Pfannenwendermanipulation erstellt. Es modelliert, dass der Ort des Smacksobjektes frei sein muss, um diese ausführen zu können. Aufgrund der mechanischen Relationen folgt daraus, dass zuvor das Vitalisobjekt entfernt werden muss. Das System plant dies korrekt. Der entstandene Plan im symbolischen Modell ist in Abb. 8.21 visualisiert. Die Ausführung ist in Abb. 8.22 dargestellt. Auf ein Ablegen des Objektes muss aus Platzgründen verzichtet werden. Stattdessen wird das Objekt in einen Karton neben dem Tisch fallen gelassen. Die Zerlegung von „Pick-and-Place" konnte dafür auf atomarer Ebene des symbolischen Modells um eine Variante erweitert werden, in der das Ablegen (Place) durch ein Öffnen der Hand ersetzt wird (Drop).

Der dritte Teil des Versuches beinhaltet den eigentlichen Transport des Toasts. Dabei wird durch das evaluierte System zur Handlungsadaption die Ausführung einer Trajektorie aus dem Mani-

pulationsplaner ausgelöst. Es existieren keine weiteren Hindernisse in der Szene, daher wird auch dieser Teil nicht weiter ausgewertet.

Der Versuch wurde über zehn mal durchgeführt. Dabei war das System zur Handlungsadaption immer in der Lage, eine geeignete Szene anzuordnen und die Ausführung der atomaren Aktionen und Trajektorien auszulösen. Die symbolische Planung für das Beiseiteräumen der Schachtel benötigte dabei ca. 5 s. Die Planungszeit profitiert im Vergleich zum „Einschenken" Versuch aus Abschnitt 8.3.3 davon, dass hier keine Ablagepose berücksichtigt werden muss. Auch die Berechnung der mechanischen Szenenrelationen ist mit 6 s schneller als bei vorigen Versuchen. Dort muss nur ein Objektpaar berücksichtigt werden. Die Zeiten sind in Tabelle 8.7 zusammengefasst.

8.4. Fazit

In diesem Kapitel wurde das entworfene und implementierte System experimentell evaluiert. Die durchgeführten Experimente sind in zwei Blöcke untergliedert. Im ersten Block wurden die einzelnen Komponenten unabhängig voneinander evaluiert. Dabei wurde gezeigt, dass die unterschiedlichen Komponenten Ergebnisse im Rahmen ihrer spezifischen Anforderungen liefern.

Die Arbeitsraumbeschreibung liefert ein kontinuierliches Maß zur Bewertung von Orten zum platzieren von Objekten. Gut bewertet Orte sind in Versuchen kinematisch erreichbar. Ihre Laufzeit spielt für das System zur Handlungsadaption keine Rolle.

Die Symbolisierung der Szenenmechanik liefert eine Beschreibung der Szene in Form eines Szenengraphens. Dieser entspricht in den Experimenten den intuitiven Erwartungen. Er kann direkt in ein symbolisches Modell abgebildet werden. Für seinen Aufbau benötigt das System bis zu 33 s, damit ist diese Komponente zusammen mit der symbolischen Planung für die online-Fähigkeit des Systems kritisch, entspricht aber den Anforderungen, eine Planungszeit von unter einer Minute zu erreichen.

Die Ausführungsüberwachung kann beim Greifen von Objekten in mehreren Versuchen mit hoher Zuverlässigkeit Fehler in der Ausführung erkennen. Auch bei ihr spielt die Laufzeit für die Onlinefähigkeit des Systems keine Rolle.

Im zweiten Block des Kapitels wurde das Gesamtsystem anhand seiner Fähigkeit, ausgewählte Manipulationsaufgaben auszuführen, evaluiert. Es wurden fünf unterschiedliche Szenen mit Transportaufgaben und spezifisch geplanten Manipulationsaufgaben betrachtet. Mit Wiederholungen wurden über 30 Versuche durchgeführt. Die vorgestellten Versuche wurden auf drei unterschiedlichen Robotern durchgeführt. Dies belegt auch die Möglichkeit, das System einfach an neue Hardware anzupassen.

Dabei konnten für die gegebenen Problemstellungen Manipulationssequenzen erzeugt werden, die die gewählten Aufgaben mit den gegebenen Anforderungen lösten. Die entwickelten Komponenten fügen sich zu einem Gesamtsystem, das online in der Lage ist, autonom zielgerichtete Manipulationssequentzen für unterschiedliche Szenen zu erzeugen. Aktionen können zu Sequenzen aneinandergereiht werden, die Aufgaben erfüllen, die für den Roboter durch einzelne Aktionen nicht lösbar sind. Das aufgebaute symbolische Modell ist in den Versuchen geeignet, die symbolische Planung zu sinnvollen Ergebnissen zu führen. Die Anbindung der Manipulationsplaner an den symbolischen Planer erlaubt die Berücksichtigung der Roboterkinematik, obwohl diese im Symbolischen nicht explizit modelliert ist.

In allen Versuchen lag die Zeit vom Beginn der Planung bis zum fertigen Plan bei weniger als einer Minute, im aufwändigsten Fall bei 49 s. Die Onlinefähigkeit gilt also auch für das Gesamtsystem. Es ist davon auszugehen, dass diese durch parallele Berechnungen noch verbessert werden kann.

9. Zusammenfassung und Ausblick

In dieser Arbeit wurde ein System zur szenenabhängigen Online-Adaption von Manipulationssequenzen für einen Serviceroboter entwickelt und evaluiert. In diesem Abschnitt werden die Ergebnisse der Arbeit zusammengefasst und diskutiert. Es wird ein Ausblick auf Erweiterungsmöglichkeiten gegeben.

9.1. Ergebnisse und Beitrag

Die Ergebnisse dieser Arbeit sind vier thematischen Blöcken zugehörig und werden hier anhand dieser strukturiert vorgestellt.

Handlungsadaption Die Adaption von Manipulationssequenzen erfordert ein zielgerichtetes Planen von Aktionen. Es wurde ein formales Metamodell entwickelt, welches den Sense-Plan-Act Zyklus von der Wahrnehmung zur Ausführung um zusätzliche Ebenen erweitert, die unterschiedlich abstrakte Darstellungen der Planung beschreiben. Als Herausforderung für diese Arbeit geht daraus die unterschiedliche Mächtigkeit von symbolischen und kontinuierlichen Modell sowie der Wirklichkeit hervor. Diese kann zum Divergieren der Modelle führen und damit in nicht zielführenden Plänen resultieren.

Aufgrund des Modells wurde ein System konzipiert, das kontinuierliche und symbolische Modelle während der Planung konsistent hält. Dies ermöglicht die Verwendung des symbolischen Planers. Es erstellt aufgrund einer wahrgenommenen Szene autonom eine symbolische Beschreibung. Während der Planung bilden Manipulationsplaner die geplanten Aktionen zurück in das kontinuierliche Modell ab. Dort wird ihre Ausführbarkeit sicher gestellt. Dabei werden semantische Annotationen des symbolischen Plans zur Parametrierung eingesetzt. Manipulationsplaner bilden die geplanten Aktionssequenzen auf ausführbare Trajektorien ab. Die Ausführung verwendet Parameter des kontinuierlichen Modells, um die Ausführung der aktuell wahrgenommenen Szene anzupassen.

Um die Konsistenz zwischen Ausführung und Plan zu sichern, ist eine Komponente zur Ausführungsüberwachung integriert. Sie ermöglicht es, im Falle von Abweichungen vom geplanten Verlauf eine Neuplanung auszulösen.

Symbolisches Modell Aufgrund der Analyse von Manipulationsaktionen wurden Aspekte identifiziert, welche vom symbolischen Modell beschrieben werden, um eine zielgerichtete Planung von Manipulationssequenzen zu ermöglichen. Es wurde ein symbolisches Modell entwickelt, das es ermöglicht, Zusammenhänge aus einer wahrgenommenen Szene in der symbolischen Planung zu berücksichtigen. Das Modell enthält die Entitäten der Orte, Objekte, Aktionen sowie der Ressourcen, die Aktionen auf dem Roboter ausführen. Manipulationsspezifisch werden insbesondere drei Aspekte beschrieben:

Arbeitsraum Ortssymbole an freien, erreichbaren Orten ermöglichen es dem Planer, Objekte an beliebigen Stellen auf einem Tisch zu platzieren. Durch ein Bewertungsmaß wird eine hohe Wahrscheinlichkeit der Ausführbarkeit im Kontinuierlichen bzw. der Wirklichkeit erreicht.

Mechanische Szenenrelationen Mit den mechanischen Szenenrelationen wurde eine Beschreibung vorgestellt, welche es ermöglicht, in der symbolischen Planung die Auswirkungen der Manipulationsaktionen auf Objekte zu berücksichtigen, die nicht Objekte der geplanten Aktion sind. Stattdessen modellieren sie Abhängigkeiten aufgrund der Statik einer Szene. Es werden die Prädikate „Stützt", „Stabil" und „Unstabil" verwendet.

Vorbedingungen generischer Aktionen Manipulationsaktionen benötigen Freiraum in der Umgebung des manipulierten Objektes. Um dies berücksichtigen zu können, werden generische Aktionen mit „Ort Frei" Vorbedingungen modelliert.

Zur Planung von parallelen Aktionen und Berücksichtigung der zeitlichen Ausdehnung von Aktionen wurden die Aktionen und Zustände des symbolischen Modells auf eine Zeitstrahldarstellung erweitert. Zur Verbesserung der Verständlichkeit und Performance des Planers wurde das Modell hierarchisch untergliedert.

Generierung des symbolischen Modells Aufgrund der Szenenabhängikeit der Handlungssequenzen muss das entworfene symbolischem Modell vom Roboter autonom und online aus der Wahrnehmung der Szene erstellt werden. Zur Erzeugung der szenenabhängigen Symbole aus dem vorigen Abschnitt wurde ein Prozess konzipiert und umgesetzt. Er beinhaltet die folgenden Komponenten:

Arbeitsraum Damit erzeugte Ortssymbole für den Roboter kinematisch erreichbar sind, wurde eine objektspezifische Bewertungsfunktion entwickelt, welche die Greifbarkeit von Objekten an Orten beschreibt. Aufgrund dieser ist die Greifbarkeitskarte definiert. Sie wird verwendet, um Orte mit einer guten Bewertung zu selektieren und damit eine hohe Wahrscheinlichkeit der Ausführbarkeit am betrachteten Ort zu erreichen. Das Arbeitsraummodell ermöglicht die Berücksichtigung des kontinuierlichen Arbeitsraumes aus der symbolischen Domäne des Planers. Das Modell konnte über die in in dieser Arbeit aufgezeigten Anwendungen auch auf andere

Problemstellungen übertragen werden, etwa zur Planung eines robotischen Wartungsvorgang eines modularen Satelliten.

Mechanische Szenenrelationen Es wurde ein Verfahren zur autonomen Erzeugung der mechanischen Szenenrelationen aufgrund von Objektlagen und -modellen entwickelt. Es basiert auf einer Physiksimulation, in der Bewegungen von Objekten und Kräfte zwischen diesen simuliert werden. Unterschiedliche Tests werden auf die so prädizierten Szenenkonfigurationen angewandt. Ihre Ergebnisse werden zu Relationen zwischen je zwei Objekten fusioniert. Es entsteht der Szenengraph, der in das symbolische Modell abgebildet wird. Das Verfahren ermöglicht die Berücksichtigung von Effekten einer Manipulationshandlung auf Objekte der Szene, die nicht selbst Objekt der Handlung sind. Dadurch werden die Möglichkeiten der Prädiktion in der klassischen symbolischen Planung in Richtung der Anwendung in realen Umgebungen verschoben.

Vorbedingungen generischer Aktionen Es wurde ein Verfahren zum Bestimmung von Vorbedingungen generischer Aktionen aus Manipulationsplanern entwickelt. Dazu werden erzeugte Trajektorien abgetastet und auf die Tischebene projiziert. Die Schnittmenge mit den erzeugten Ortssymbolen beschreibt die notwendigen Vorbedingungen der Aktion. Ihre Einhaltung erlaubt die kollisionsfreie Ausführung. Eine solche Einbindung von Verfahren auf kontinuierlichen Modellen an die symbolische Planungsdomäne geht deutlich über bekannte Methoden des Standes der Technik hinaus und ermöglicht die Berücksichtigung beliebiger Manipulationsaktionen bei der Planung von Handlungssequenzen.

Ausführungsüberwachung Die entwickelte Ausführungsüberwachung erlaubt es, Abweichungen von der geplanten Ausführung auf Aktionsebene in Echtzeit zu detektieren. Mittels überwachten Lernens wurde eine Support Vector Machine auf Sensordaten trainiert. Der Benutzer annotiert dazu Ausführungen bezüglich ihres Erfolges. Während der Ausführung wird zu jeder Zeit aufgrund von Sensorwerten eine Klassifikation des aktuellen Zustandes in „korrekt" oder „fehlerhaft" durchgeführt. Damit können Fehler frühzeitig erkannt werden. Das Anhalten der aktuellen Bewegung verhindert Schäden an Roboter und Objekten; durch Neuplanung kann ein konsistenter Plan erzeugt werden.

9.2. Diskussion

Das vorgestellte System und seine Komponenten wurden in Experimenten evaluiert. Dabei war es in der Lage, die Anforderungen der szenenabhängigen Online-Adaption von Manipulationssequenzen zu erfüllen. In zahlreichen Versuchen mit dem Gesamtsystem konnten unterschiedliche, exemplarisch ausgewählte Szenen so angepasst werden, dass angefragte Aufgaben ausgeführt werden konnten. Im Stand der Technik ist kein anderes System bekannt, das zum eine

so hohe Flexibilität von Planung und Modell auf abstrakter Ebene bietet und gleichzeitig die Ausführbarkeit von Manipulationsaktionen auf realen Robotern sicherstellt. Zu dieser Fähigkeit trägt insbesondere das entwickelte symbolische Modell und seine automatische Erzeugung aufgrund der Wahrgenommenen Szene bei. Die erreichte Tiefe der Modellierung von Zusammenhänge zwischen Objekten in der Szene geht über den Stand der Technik hinaus. Das System kann damit eine Handlungssequenz unterschiedlichen Szenen anpassen. Autonomie und Ausführungszeiten der Modellerstellung führen zur Onlinefähigkeit des Systems.

Die Nutzung eines symbolischen Planers erlaubt es, auch komplexe Zusammenhänge zwischen Aktionen und Objekten zu modellieren und dafür zielführende Aktionen zu planen. Die erstellten Pläne haben einen klaren Bezug zu ihrer Semantik, die für den Benutzer leicht zu erfassen ist. Aufgrund der im Planer verwendeten Zeitstrahldarstellung ist das System in der Lage, Aktionen geeigneten Ressourcen zuzuordnen und damit effiziente Pläne zu erstellen.

Auch ist es konzeptionell einfach, weiteres Wissen in die Planung einfließen zu lassen, da neue Prädikate und Bedingungen unabhängig in das symbolische Modell eingefügt werden können. Dies ermöglicht auch das zusätzliche Einbringen von Expertenwissen. Mit der verwendeten Implementierung der symbolischen Planung gilt dies jedoch unter Einschränkungen. Zusätzliche Symbole beeinflussen den Zeitbedarf der Planung unter Umständen negativ. Änderungen am symbolischen Modell müssen vom Experten unter Berücksichtigung dieses Umstands optimiert werden.

Durch die abstrakte Modellierung der Manipulation lässt sich das System leicht auf andere Roboterkonfigurationen übertragen. In den Experimenten wurde dies an fünf verschiedenen Roboterkonfigurationen gezeigt. Auch neue Aktionen können aufgrund der generischen Freiraumbedinungen und Integration der Manipulationsplaner einfach hinzugefügt werden.

Die Aktionsplanung erlaubt es, die Risiken durch Kollisionen bei robotischer Manipulation zu reduzieren, da Trajektorien in der Nähe von Hindernissen zugunsten einer Veränderung der Szene verworfen werden können. Auch kann die Effizienz der Manipulation erhöht werden, wenn Aktionen gleichzeitig durch mehrere Roboterkomponenten ausgeführt werden.

Die erreichte Flexibilität ist für einen autonomen Serviceroboter eine Grundvoraussetzung. Sie kann jedoch auch in heutigen industriellen robotischen Anwendungen Vorteile bringen. So könnte das Verfahren angepasst werden, um Objekte, die als Schüttgut vorliegen zu palettieren oder einzeln weiter zu verarbeiten. Mit geeigneten symbolischen Modellen kann die Planung von (De-)Montagevorgängen für einen Roboter vorgenommen werden.

9.3. Ausblick

Die Laufzeit des umgesetzten Verfahrens hängt vom symbolischen Modell und Planer ab. Um diese zu stabilisieren, erscheint eine domänenspezifische Planung erfolgversprechend. Dabei handelt es sich um einen hierarchischen Planer, bei dem zusätzliches Aufgabenwissen und Heuristiken zur Zerlegung von Aktionen eingebracht werden. In dieser Arbeit wurden diese nicht betrachtet, da Nebenläufigkeit und die Verteilung auf Ressourcen dort nicht darstellbar sind. Es sollte untersucht werden, ob ein hierarchisch geplanter Plan um diese ergänzt werden kann. Alternativ wären Planungsverfahren erstrebenswert, die ihre Suchstrategie selbstständig an die Planungsdomäne anpassen und optimieren können. Dies ermöglicht es, die Planungsdomäne jederzeit anpassen zu können, insbesondere auch online durch einen selbst lernenden Roboter.

Die entwickelte Ausführungsüberwachung ist in der Lage, fehlerhafte Zustände zu erkennen und eine Neuplanung auszulösen. Werden die Fehler feingranularer erkannt, so kann das Wissen über die Art des Fehlers bei der Reaktion darauf genutzt werden. Es könnte darauf verzichtet werden, vollständig neu zu planen. Werden geeignete Erwartungen aus dem symbolischen Plan erzeugt, kann auch erkannt werden, dass ein Fehlerzustand für die Ausführung irrelevant ist. Die Neuplanung muss dann nicht durchgeführt werden. Insgesamt kann das System mit dieser Erweiterung schneller auf Abweichungen der Ausführung reagieren.

Das Verfahren zur Erzeugung der mechanischen Szenenrelationen kann um eine Prädiktion der Ergebnisses von Manipulationsaktionen auf Szenen aus verrauschter Sensorik erweitert werden. Damit kann bewertet werden, wie wahrscheinlich eine Aktionssequenz zum symbolisch geplanten Ergebnis führt. Der Planer plant nur bis zu einem Horizont im Voraus, in dem seine Ergebnisse mit hoher Wahrscheinlichkeit mit der Realität konsistent sind; aufgrund der tatsächlichen Wahrnehmung wird der Plan während der Ausführung aktualisiert.

Der größte Verbesserungspunkt auf dem Weg zum autonomen Serviceroboter im Haushalt oder auch in der Produktion ist zum Ende dieser Arbeit die Abbildung von kollisionsbehafteten Manipulationsaktionen. Dort mangelt es dem Stand der Technik an vielseitig anwendbaren Verfahren. Neben den geometrischen Eigenschaften der Manipulation müssen hier auch Kräfte berücksichtigt werden, ebenso wie die Eigerenschaften der manipulierten Objekte.

In der Evaluation wurde die Integration in eine Missionssteuerung betrachtet. Diese ist notwendig, um Aspekte jenseits der Manipulation berücksichtigen zu können. Die bekannten Ansätze besitzen jedoch nicht die hohe Flexibilität bezüglich der Umgebung, sondern werden spezifisch für Szenarien trainiert. Für einen autonomen Serviceroboter besteht auch hier weiterer Entwicklungsbedarf.

A. Überblick Planungsverfahren in der Robotik

In diesem Anhang werden Verfahren untersucht, die Aktionen für Service Roboter erzeugen.

Nach (Siegert u. Bocionek, 1996) bedeutet Planung: „[. . .] zu einer definierten Aufgabe deren Durchführung zu bestimmen". Dazu sind in der Literatur eine Vielzahl an Verfahren bekannt. Die Aufgabe wird meist als Ziel betrachtet, es soll ein Plan erzeugt werden, der dieses erreicht. Sie unterscheiden sich in der Art der definierten Aufgabe, die zu fundamental unterschiedlichen Perspektiven auf die Beschreibung der erzeugten Durchführung führt. In diesem Abschnitt wird eine Taxonomie unterschiedlicher, für die Robotik relevanter Planungsverfahren vorgestellt. In der Taxonomie werden Zweige identifiziert, die für die Problemstellung dieser Arbeit von Bedeutung sind. Sie werden in den folgenden Abschnitten genauer betrachtet. Die Systematik wurde anhand von in (Siciliano, B. and Khatib, 2008) aufgeführten Verfahren entwickelt; die Auswahl der beschriebenen und eingeordneten Verfahren ist von der Quelle unabhängig.

Die entworfene Taxonomie ist in Abb. A.1 dargestellt. Zu den Blättern der Taxonomie gibt es Verfahren, die sich ausschließlich auf das Problem des spezifizierten Zweiges beschränken. In der Robotik kommen die geschilderten Problemstellungen selten losgelöst voneinander vor, daher sind in der Literatur eine Vielzahl von hybriden Ansätzen zu finden, die sich nicht klar einem einzelnem Blatt zuordnen lassen.

Auf oberster Ebene unterscheidet die Taxonomie zwischen kontinuierlichen und diskreten Planungsproblemen. Dabei wird die zeitliche Kontinuität der Lösung betrachtet. Verfahren, die als kontinuierlich eingeordnet werden, erzeugen für jede geplante Eigenschaft der Lösung zu jedem Zeitpunkt in einem angestrebten Lösungszeitraum einen Wert. Die Sequenz der Lösungen wird als Trajektorie bezeichnet. Je nach Problemstellung kann sie beispielsweise im Gelenkwinkelraum des Roboters oder in sechsdimensionalen kartesischen Lagekoordinaten definiert sein. Im Gegensatz dazu erzeugen Planer zu als diskret eingeordneten Planungsproblemen eine endliche Menge Ergebnisse. Diese können einen zeitlichen Bezug haben. In der klassischen Aktionsplanung etwa ist das Ergebnis durch eine Abfolge von Symbolen definiert.

Im kontinuierlichen Zweig wird auf der nächsten Ebene unterschieden, ob es sich um eine kollisionsfreie Planung oder eine kontaktbehaftete Planung handelt. Im Falle der kollisionsfreien Planer handelt es sich um Bahnplaner, die den Roboter oder Teile des Roboters an einen anderen Ort bewegen, jedoch die Umwelt nicht verändern. Es wird nur der Zustand des Ro-

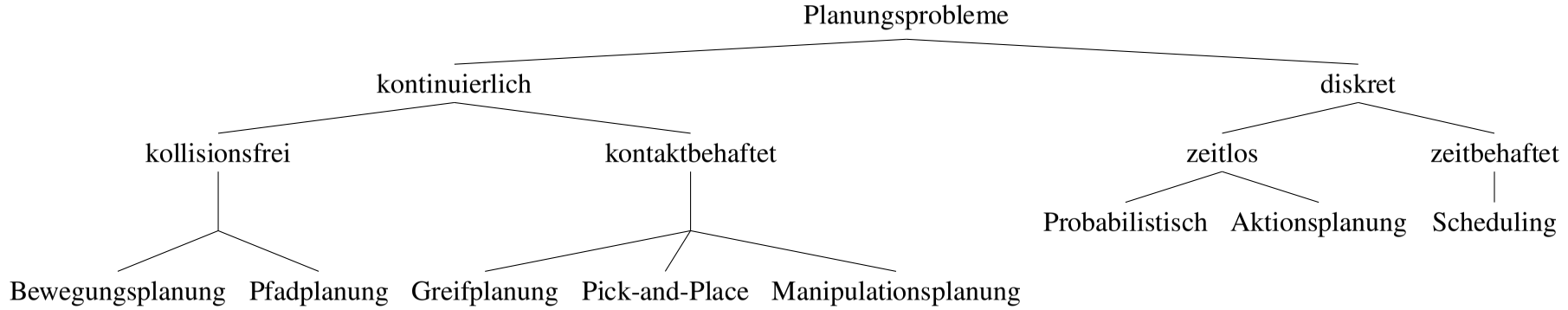


Abb. A.1.: Taxonomie unterschiedlicher Planungsprobleme. Unter kontinuierlich eingeordnete Probleme sind auf Trajektorien bestimmt. Ihr Ursprung liegt meist in der Robotik. Die diskreten Probleme sind auf Ereignissen und Aktionen definiert, sie entstammen eher dem Bereich der künstlichen Intelligenz.

boters verändert. Eben dies wird durch die Kontaktlosigkeit erreicht. Sie lasen sich weiter in Bewegungsplanungs- und Pfadplanungsverfahren unterteilen.

Pfadplanungsverfahren [1] (Minguez u. a., 2008) betrachten den gesamten Roboter, der sich beispielsweise mittels einer mobilen Plattform im Raum bewegt. Ein Zustand des Roboters im Planer, die sog. Roboterkonfiguration wird dabei meist im kartesischen Raum festgelegt.

Bei der Bewegungsplanung dagegen stehen die kinematischen Zwänge im Vordergrund, die durch die meist serielle Struktur eines Manipulators gegeben sind. Dessen Zustand ist durch die Konfiguration seiner Gelenke festgelegt, bei einer Kinematik mit ausschließlich rotatorischen Freiheitsgraden im Gelenkwinkelraum. Einen Überblick über Bewegungsplanungsverfahren[2] wird in (Kavraki u. LaValle, 2008; Hwang u. Ahuja, 1992) gegeben.

Die Problematik von Pfad- und Bewegungsplanung ähnelt sich so weit, dass eine Vielzahl der in der Literatur bekannten Verfahren für beide Probleme eingesetzt wird. Eine Unterscheidung ist oft nur durch die Motivation des Autors gegeben. In (Hwang u. Ahuja, 1992) wird eine weitere Unterteilung von Bewegungsplanungsverfahren vorgeschlagen. Sie nutzt Eigenschaften wie Vollständigkeit und Gültigkeit. Damit zielt sie stark auf die Methodik des verwendeten Algorithmus ab und nicht auf die Charakteristik des Planungsproblems. In dieser Arbeit wird sie daher nicht verwendet.

Im zweiten Zweig, der kontaktbehafteten Planung, befinden sich Aufgaben, in denen die Umwelt durch das Agieren des Roboters verändert wird. Dazu ist es notwendig, mechanischen Kontakt zwischen dem Roboter und manipulierten Objekten herzustellen. Viele Verfahren in diesem Gebiet basieren auf Bewegungsplanungsverfahren (Brock u. a., 2008).

Bei der kontaktbehafteten Planung unterscheidet diese Arbeit zwischen Greifplanung, Pick-and-Place Planung und weiteren Manipulationsplanern. Bei der Greifplanung soll ein Kontakt zwischen Roboter, insbesondere des Greifers des Roboters und einem Objekt derart hergestellt werden, dass das Objekt möglichst fest mit dem Roboter verbunden ist und von diesem transportiert werden kann. Diese spielt als verwendete Grundlage für diese Arbeit eine wichtige Rolle und wird in Abschnitt 2.4.2 behandelt.

Das Problem Aktionen für mehrere, bewegliche Objekte zu erzeugen wird in der Literatur als Manipulationsplanung (LaValle, 2006), bzw. wenn nur der Raum der Objekte und kein Manipulator bertrachtet wird als Rearrangier-Planung (Rivlin, 1998) bezeichnet. Es ist PSPACE hart (Wilfong, 1988). In dieser Arbeit wird der Name Manipulationsplanung nicht in diesem Sinne verwendet, sondern bezeichnet die Planung einer Aktion eines Roboters, durch die gezielt Objekte durch das Ausüben von Kräften beeinflusst werden. Stattdessen wird der Begriff der

[1] engl.: obstacle avoidance
[2] engl.: motion planning

Pick-and-Place-Planung verwendet. Dies ist ein Forschungsbereich, zu dem diese Arbeit einen Beitrag leistet, der Stand der Technik wird in Abschnitt 3.2 behandelt.

In den Bereich der weiteren Manipulationsplaner fallen Planer, die Kontakte zwischen Roboter und einzelnen Punkten der Objektes herstellen, oft detaillierter unter Berücksichtigung auftretender Kräfte und Kontakten zu weiteren Objekten. Dazu zählt vor allem die feinmotorische Manipulation[3] (Trinkle u. Hunter, 1991; Rus, 1992), sowie die Planung von Verschiebeaktionen (Lau u. a., 2011). Ein weiteres relevantes Gebiet ist die Planung von Montageabläufen[4] (Ewert u. a., 2010). Dabei wird von einem gegriffenen Objekt ausgegangen, es müssen jedoch weiter Randbedingungen beim Zusammenfügen von Objekten berücksichtigt werden; etwa beim Einführen eines Bolzens in ein Loch[5](Mason, 1981).

[3]engl.: dexterous manipulation
[4]engl.: assmbly planning
[5]engl.: peg in hole problem

Abbildungsverzeichnis

Tabellenverzeichnis

Literatur

[Abdel-Malek u. a. 2004] ABDEL-MALEK, K ; YU, W ; YANG, J: Placement of Robot Manipulators to Maximize Dexterity. In: *International Journal of Robotics and Automation* 19 (2004), Nr. 1, S. 615

[Allen 1983] ALLEN, JF F.: Maintaining Knowledge about Temporal Intervals. In: *Communications of the ACM* 26 (1983), November, Nr. 11, 832–843. `http://portal.acm.org/citation.cfm?id=358434`

[Ambros-Ingerson u. Steel 1988] AMBROS-INGERSON, J. A. ; STEEL, S.: Integrating Planning, Execution and Monitoring. In: *Proceedings of the Seventh National Conference on Artificial Intelligence*. Saint Paul, Minnesota, 1988, 83–88

[Aydin u. Nakajima 1999] AYDIN, Yahya ; NAKAJIMA, Masayuki: Database Guided Computer Animation of Human Grasping Using Forward and Inverse Kinematics. In: *Computers & Graphics* 23 (1999), Nr. 1, 145–154. `http://linkinghub.elsevier.com/retrieve/pii/S0097849398001228`. – ISSN 0097–8493

[Balbiani u. a. 1999] BALBIANI, Philippe ; CONDOTTA, Jean-François ; DEL CERRO, Luis F.: A new tractable subclass of the rectangle algebra. In: *Proceedings of the 16th international joint conference on Artifical intelligence (IJCAI-99)* Bd. 1, 1999. – ISBN 1–55860–613–0, S. 442–447

[Becher 2008] BECHER, R: *Semantische Objektmodellierung mittels multimodaler Interaktion*, Karlsruher Institut für Technologie (KIT), Dissertation, 2008. `http://digbib.ubka.uni-karlsruhe.de/volltexte/1000009371`

[Bedrax-Weiss u. a. 2004] BEDRAX-WEISS, T ; FRANK, J ; JÓNSSON, A ; MCGANN, C: EUROPA2: Plan Database Services for Planning and Scheduling Applications. Version: 2004. `http://ntrs.nasa.gov/search.jsp?R=20050157874`. 2004. – Forschungsbericht

[Beetz u. a. 2012] BEETZ, Michael ; JAIN, Dominik ; MÖSENLECHNER, Lorenz ; KUNZE, Lars ; BLODOW, Nico ; PANGERCIC, Dejan: Cognition-Enabled Autonomous Robot Control for the Realization of Home Chore Task Intelligence. In: *Proceedings of the IEEE, Special Issue*

on Quality of Life Technology 100 (2012), Nr. 8, 2454–2471. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6235978

[Berenson u. a. 2007] BERENSON, Dmitry ; DIANKOV, Rosen ; KUFFNER, James: Grasp Planning in Complex Scenes. In: *2007 7th IEEE-RAS International Conference on Humanoid Robots* (2007), November, 42–48. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4813847. ISBN 978–1–4244–1861–9

[Berenson u. a. 2008] BERENSON, Dmitry ; KUFFNER, James ; CHOSET, Howie: An Optimization Approach to Planning for Mobile Manipulation. In: *IEEE International Conference on Robotics and Automation (ICRA) 2008* Bd. 2008, IEEE, Mai 2008. – ISBN 978–1–4244–1646–2, 1187–1192

[Berenson u. a. 2011] BERENSON, Dmitry ; SRINIVASA, S. ; KUFFNER, James: Task Space Regions: A Framework for Pose-Constrained Manipulation Planning. In: *The International Journal of Robotics Research* 30 (2011), März, Nr. 12, 1435–1460. http://ijr.sagepub.com/cgi/doi/10.1177/0278364910396389. – ISSN 0278–3649

[Bernardini u. Smith 2007] BERNARDINI, Sara ; SMITH, D.E.: Developing Domain-Independent Search Control for EUROPA2. In: *ICAPS Workshop on Heuristics for Domain-independent Planing* Bd. 1. Providence, Rhode Island, USA, 2007

[Bischoff u. Kurth 2010] BISCHOFF, Rainer ; KURTH, Johannes: The Kuka-DLR Lightweight Robot Arm - a New Reference Platform for Robotics Research and Manufacturing. In: *Robotics (ISR), 2010 41st International Symposium on and 2010 6th German Conference on Robotics (ROBOTIK)*. Munich, Germany, 2010. – ISBN 9783800732739, 741–748

[Bogoni u. Bajcsy 1995] BOGONI, L ; BAJCSY, R: Interactive Recognition and Representation of Functionality. In: *Computer Vision and Image Understanding* 62 (1995), S. 194–214. – ISBN 1077–3142

[Brock u. a. 2008] BROCK, Oliver ; KUFFNER, James ; XIAO, Jing: Motion for Manipulation Tasks. In: SICILIANO, BRUNO, KHATIB, Oussama (Hrsg.): *Springer Handbook of Robotics*. Heidelberg : Springer Berlin Heidelberg, 2008, S. 615–645

[Bullet 2014] *Bullet Real-Time Physics Simulation*. http://bulletphysics.org, 2014

[Burbridge u. Dearden 2012] BURBRIDGE, Chris ; DEARDEN, Richard: Learning the Geometric Meaning of Symbolic Abstractions for Manipulation Planning. Version: 2012. http://dx.doi.org/10.1007/978-3-642-32527-4_20. In: HERRMANN, Guido (Hrsg.) ; STUDLEY, Matthew (Hrsg.) ; PEARSON, Martin (Hrsg.) ; CONN, Andrew (Hrsg.) ; MELHUISH, Chris (Hrsg.) ; WITKOWSKI, Mark (Hrsg.) ; KIM, Jong-Hwan (Hrsg.) ; VADAKKEPAT, Prahlad

(Hrsg.): *Advances in Autonomous Robotics* Bd. 7429. Springer Berlin Heidelberg, 2012. – ISBN 978–3–642–32526–7, 220–231

[Cambon u. a. 2009] CAMBON, S. ; ALAMI, Rachid ; GRAVOT, Fabien: A Hybrid Approach to Intricate Motion, Manipulation and Task Planning. In: *The International Journal of Robotics Research* 28 (2009), S. 104–126. – ISSN 0278–3649

[Cambon u. a. 2004] CAMBON, S ; GRAVOT, Fabien ; ALAMI, Rachid: A Robot Task Planner that Merges Symbolic and Geometric Reasoning. In: *European Conference on Artificial Intelligence*. Valencia, 2004, 895–899

[Chang u. Lin 2001] CHANG, Chih-Chung ; LIN, Chih-Jen: *LIBSVM: a Library for Support Vector Machines*. 2001

[Choi u. Amir 2009] CHOI, Jaesik ; AMIR, Eyal: Combining Planning and Motion Planning. In: *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. Piscataway, NJ, USA : IEEE, 2009. – ISBN 978–1–4244–2788–8, 238–244

[Clark 1973] CLARK, Herbert H.: Space, Time, Semantics, and the Child. In: MOORE, Timothy E. (Hrsg.): *Cognitive Development and the Acquisition of Language*. Academic Press, 1973. – ISBN 978–0125058506

[Cortes u. Vapnik 1995] CORTES, Corinna ; VAPNIK, Vladimir: Support-Vector networks. In: *Machine Learning* 20 (1995), September, Nr. 3, 273–297. `http://www.springerlink.com/index/10.1007/BF00994018`. – ISSN 0885–6125

[Daley u. a. 2005] DALEY, Patrick ; FRANK, Jeremy ; IATAURO, Michael ; MCGANN, Conor ; W ; TAYLOR, Will: Planworks: A Debugging Environment for Constraint Based Planning Systems. Monterey, California, USA, 2005. – Forschungsbericht. – 22–26 S.

[Dearden u. a. 2004] DEARDEN, Richard ; WILLEKE, Thomas ; HUTTER, Frank ; SIMMONS, Reid ; VERMA, Vandi ; THRUN, Sebastian: Real-Time Fault Fetection and Situational Awareness for Rovers: Report on the Mars Technology Program ask. In: *In Proceedings of IEEE Aerospace Conference*, 2004. – ISBN 0780381556, 826–840

[Diosi u. a. 2005] DIOSI, A. ; TAYLOR, G. ; KLEEMAN, L.: Interactive SLAM using Laser and Advanced Sonar. In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation* (2005). ISBN 0–7803–8914–X

[Dornhege u. a. 2010] DORNHEGE, Christian ; GISSLER, Marc ; TESCHNER, Matthias ; NEBEL, Bernhard: Integrating Symbolic and Geometric Planning for Mobile Manipulation. In: *Safety, Security & Rescue Robotics (SSRR), 2009 IEEE International Workshop on*, IEEE, November 2010, 1–6

[Dornhege u. Hertle 2013] DORNHEGE, Christian ; HERTLE, Andreas: Integrated Symbolic Planning in the Tidyup-Robot Project. In: *AAAI Spring Symposium - Designing Intelligent Robots: Reintegrating AI II*, 2013

[Dornhege u. a. 2013] DORNHEGE, Christian ; HERTLE, Andreas ; NEBEL, Bernhard: Lazy Evaluation and Subsumption Caching for Search-Based Integrated Task and Motion Planning. In: *Proceedings of the IROS workshop on AI-based robotics*, 2013

[Duda u. a. 2001] DUDA, Richard O. ; HART, Peter E. ; STORK, David G.: *Pattern Classification*. 2. ed. New York : Wiley, 2001. – ISBN 0–471–05669–3 ; 978–0–471–05669–0

[Ehrenmann 2003] EHRENMANN, Markus: *Handlungsbeobachtung zur Instruierung von Robotersystemen*, Unversität Karlsruhe, Dissertation, 2003

[Ewert u. a. 2010] EWERT, Daniel ; THELEN, Sebastian ; KUNZE, Ralph ; MAYER, Marcel ; SCHILBERG, Daniel ; JESCHKE, Sabina: A Graph Based Hybrid Approach of Offline Pre-Planning and Online Re-Planning for Efficient Assembly under Realtime Constraints. In: *Intelligent Robotics and Applications* (2010), 44–55. http://www.springerlink.com/index/W85100632M08160K.pdf

[Fedrizzi u. a. 2009] FEDRIZZI, Andreas ; MOESENLECHNER, L. ; STULP, Freek ; BEETZ, Michael: Transformational Planning for Mobile Manipulation Based on Action-Related Places. In: *Advanced Robotics, 2009. ICAR 2009. International Conference on*, IEEE, 2009

[Fikes u. Nilsson 1971] FIKES, Richard E. ; NILSSON, Nils J.: STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. In: *Artificial Intelligence* 2 (1971), Nr. 3-4, 189–208. http://linkinghub.elsevier.com/retrieve/pii/0004370271900105. – ISSN 00043702

[Frank u. Jonsson 2002] FRANK, Jeremy ; JONSSON, Ari: Constraint-Based Attribute and Interval Planning / NASA Ames Research Center. Version: 2002. http://ti.arc.nasa.gov/m/pub-archive/313h/0313(Frank).pdf. 2002. – Forschungsbericht. – 1–32 S.

[Freeman 1975] FREEMAN, J: The Modelling of Spatial Relations. In: *Computer Graphics and Image Processing* 4 (1975), Juni, Nr. 2, 156–171. http://linkinghub.elsevier.com/retrieve/pii/S0146664X75800074. – ISSN 0146664X

[Friedrich 1999] FRIEDRICH, Holger: *Interaktive Programmierung von Manipulationssequenzen*. Herdecke, Universität Karlsruhe, Dissertation, 1999

[Gat u. a. 1990] GAT, E. ; SLACK, M.G. ; MILLER, D.P. ; FIRBY, R.J.: PathPlanning and Execution Monitoring for a Planetary Rover. In: *Proceedings., IEEE International Conference on Robotics and Automation*, IEEE Comput. Soc. Press, 1990. – ISBN 0–8186–9061–5, 20–25

[Gat 1997] GAT, Erann: On Three-Layer Architectures. In: *Artificial intelligence and mobile robots* (1997), 195–210. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.35.2130&rep=rep1&type=pdf

[Goldfeder u. a. 2007] GOLDFEDER, C. ; ALLEN, P.K. ; LACKNER, C. ; PELOSSOF, R.: Grasp Planning via Decomposition Trees. In: *Proceedings 2007 IEEE International Conference on Robotics and Automation* (2007). – ISBN 1–4244–0601–3

[Goos 2005] GOOS, Gerhard: *Vorlesungen über Informatik*. 4. 1 : Springer, 2005

[Gravot u. a. 2005] GRAVOT, Fabien ; CAMBON, Stephane ; ALAMI, Rachid: aSyMov: A Planner that Deals with Intricate Symbolic and Geometric Problems. Version: 2005. http://www.springerlink.com/index/975rpk39wew7nr4r.pdf$\delimiter"026E30F$nhttp://dx.doi.org/10.1007/11008941_11. In: *Robotics Research* Bd. 15. 2005. – ISBN 978–3–540–23214–8, 100–110

[Gravot u. a. 2006] GRAVOT, Fabien ; HANEDA, A. ; OKADA, K. ; INABA, M.: Cooking for Humanoid Robot, a Task that Needs Symbolic and Geometric Reasonings. In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* (2006). – ISBN 0–7803–9505–0

[Grundmann u. a. 2010] GRUNDMANN, Thilo ; EIDENBERGER, Robert ; SCHNEIDER, Martin ; FIEGERT, Michael ; WICHERT, G.: Robust high precision 6D pose determination in complex environments for robotic manipulation. In: *Proc. Workshop Best Practice in 3D Perception and Modeling for Mobile Manipulation at the Int. Conf. Robotics and Automation*, 2010, 1–6

[Guilamo u. a. 2005] GUILAMO, L ; KUFFNER, J ; NISHIWAKI, K ; KAGAMI, S ; S: Efficient prioritized inverse kinematic solutions for redundant manipulators. In: *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on* Bd. I, 2005, 3921–3926

[Guitton u. Farges 2009] GUITTON, Julien ; FARGES, JL: Taking into Account Geometric Constraints for Task-Oriented Motion Planning. In: *BTAMP'09 (ICAPS Workshop)*, 2009

[Hähnel u. a. 1998] HÄHNEL, Dirk ; BURGARD, Wolfram ; LAKEMEYER, Gerhard: *GOLEX — Bridging the Gap between Logic (GOLOG) and a Real Robot*. 1998

[Harnad 1990] HARNAD, Stevan: The Symbol Grounding Problem. In: *Physica D: Nonlinear Phenomena* 42 (1990), 335–346. http://groups.lis.illinois.edu/amag/langev/paper/harnad90theSymbol.html

[Hertzberg u. Chatila 2008] HERTZBERG, Joachim ; CHATILA, Raja: AI Reasoning Methods for Robotics. In: SICILIANO, BRUNO, KHATIB, Oussama (Hrsg.): *Springer Handbook of Robotics*. 2008, S. 207–223

[Hwang u. Ahuja 1992] HWANG, Y.K. ; AHUJA, N.: Gross Motion Planning - A Survey. In: *ACM Computing Surveys (CSUR)* 24 (1992), Nr. 3, 219–291. http://dl.acm.org/citation.cfm?id=136037

[IPC 2014] *The International Planning Competition*. http://ipc.icaps-conference.org/, 2014

[Isermann 2004] ISERMANN, Rolf: Model-Based Fault Detection and Diagnosis: Status and Applications. In: *Annual Reviews in Control* Bd. 29, 2004, S. 71–85

[Jäkel 2013] JÄKEL, Rainer: *Learning of Generalized Manipulation Strategies in Service Robotics*, Karlsruher Instituts für Technologie (KIT), Diss., 2013. http://digbib.ubka.uni-karlsruhe.de/volltexte/1000033899

[Jäkel u. a. 2010] JÄKEL, Rainer ; SCHMIDT-ROHR, Sven R. ; LOESCH, Martin ; DILLMANN, Ruediger: Learning of Probabilistic Grasping Strategies using Programming by Demonstration. In: *2010 IEEE International Conference on Robotics and Automation* Bd. 2010. Anchorage, USA : IEEE, Mai 2010. – ISBN 978–1–4244–5038–1, 873–880

[Jäkel u. a. 2012] JÄKEL, Rainer ; SCHMIDT-ROHR, SvenR. R. ; RÜHL, Steffen W. ; KASPER, Alexander ; XUE, Zhixing ; DILLMANN, Rüdiger: Learning of Planning Models for Dexterous Manipulation Based on Human Demonstrations. In: *International Journal of Social Robotics* 4 (2012), Nr. 4, 437–448. http://dx.doi.org/10.1007/s12369-012-0162-y. – ISSN 1875–4791

[Kaelbling u. Lozano-Perez 2010] KAELBLING, LP ; LOZANO-PEREZ, Thomas: Hierarchical Task and Motion Planning in the Now. In: *IEEE Intl. Conf. on Robotics and Automation*. Anchorage, Alaska, 2010

[Karlsson u. a. 2012] KARLSSON, Lars ; BIDOT, Julien ; LAGRIFFOUL, Fabien: Combining Task and Path Planning for a Humanoid Two-Arm Robotic System. In: *TAMPRA workshop, ICAPS 2012*, 2012

[Kasper u. a. 2012] KASPER, A. ; XUE, Z. ; DILLMANN, R.: The KIT Object Models Database: An Object Model Database for Object Recognition, Localization and Manipulation in Service Robotics. In: *The International Journal of Robotics Research* 31 (2012), Mai, Nr. 8, 927–934. http://ijr.sagepub.com/cgi/doi/10.1177/0278364912445831. – ISSN 0278–3649

[Kasper 2013] KASPER, Alexander: *Szenen- und Objektmodellierung für Serviceroboter*, Karlsruher Institut für Technologie, Diss., 2013

[Kavraki u. LaValle 2008] KAVRAKI, Lydia E. ; LAVALLE, Steven M.: Motion Planning. In: SICILIANO, BRUNO, KHATIB, Oussama (Hrsg.): *Springer Handbook of Robotics*. Heidelberg : Springer Berlin Heidelberg, 2008, Kapitel 5, S. 109–131

[Kavraki u. a. 1996] KAVRAKI, Lydia.E. ; SVESTKA, P. ; LATOMBE, J.-C. ; OVERMARS, M.H.: Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces. In: *IEEE Transactions on Robotics and Automation* 12 (1996). – ISBN 1042–296X

[Kawabata u. a. 2002] KAWABATA, K. ; AKAMATSU, T. ; ASAMA, H.: A Study of Self-Diagnosis System of an Autonomous Mobile Robot: Expansion of State Sensory Systems. In: *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on* Bd. 2. Lausanne, Schweiz : Ieee, 2002. – ISBN 0–7803–7398–7, 1802 – 1807

[Keller u. Wang 1996] KELLER, J M. ; WANG, Xiaomei: Learning Spatial Relationships in Computer Cision. In: *Proceedings of the Fifth IEEE International Conference on Fuzzy Systems* Bd. 1. New Orleans, Louisiana, 1996, S. 118–124

[Kirkpatrick u. a. 1992] KIRKPATRICK, David ; MISHRA, Bhubaneswar ; YAP, Chee-Keng: Quantitative Steinitz's Theorems with Applications to Multifingered Grasping. In: *Discrete & Computational Geometry* 7 (1992), Nr. 1, 295–318. `http://dx.doi.org/10.1007/BF02187843`. – ISSN 0179–5376

[Kleer u. Brown 1984] KLEER, Johan D. ; BROWN, John S.: A Qualitative Physics Confluences. In: *Artificial Intelligence* 24 (1984), S. 7–83

[Knoop 2007] KNOOP, Steffen: *Interaktive Erstellung und Ausführung von Handlungswissen für einen Serviceroboter*, Universität Karlsruhe, Diss., 2007

[Knoop u. a. 2006] KNOOP, Steffen ; SCHMIDT-ROHR, S.R. ; DILLMANN, R.: A Flexible Task Knowledge Representation for Service Robots. In: *The 9th International Conference on Intelligent Autonomous Systems (IAS-9), Kashiwa New Campus, The University of Tokyo, Tokyo, Japan*, Citeseer, 2006

[Kresse u. Beetz 2012] KRESSE, Ingo ; BEETZ, Michael: Movement-Aware Action Control—Integrating Symbolic and Control-Theoretic Action Execution. In: *IEEE International Conference on Robotics and Automation (ICRA)*. St. Paul, Minnesota, 2012. – ISBN 9781467314053, 3245–3251

[Kühnle u. a. 2009] KÜHNLE, Jens ; VERL, Alexander ; XUE, Zhixing ; RÜHL, Steffen W. ; ZÖLLNER, J.M. M. ; DILLMANN, Ruediger ; GRUNDMANN, Thilo ; EIDENBERGER, Robert

; ZÖLLNER, R.D. Raoul D. ; DILLMAN, Rüdiger: 6D Object Localization and Obstacle Detection for Collision-Free Manipulation with a Mobile Service Robot. In: *14th International Conference on Advanced Robotics (ICAR)*. Munich : IEEE, 2009, 1–6

[Kuipers 2000] KUIPERS, Benjamin: The Spatial Semantic Hierarchy. In: *Artificial Intelligence* 119 (2000), Nr. April 1999, 191–233. http://www.sciencedirect.com/science/article/pii/S0004370200000175

[Lau u. a. 2011] LAU, Manfred ; MITANI, Jun ; IGARASHI, Takeo: Automatic Learning of Pushing Strategy for Delivery of Irregular-Shaped Objects. In: *2011 IEEE International Conference on Robotics and Automation*. Shanghai, China : IEEE, Mai 2011. – ISBN 978–1–61284–386–5, 3733–3738

[LaValle 1998] LAVALLE, S M.: Rapidly-Exploring Random Trees: A New Tool for Path Planning. In: *In* 129 (1998), S. 98–11

[LaValle u. Kuffner 1999] LAVALLE, S.M. ; KUFFNER, J.J.: Randomized Kinodynamic Planning. In: *Proceedings 1999 IEEE International Conference on Robotics and Automation* Bd. 1. Detroit, Michigan, 1999. – ISBN 0–7803–5180–0

[LaValle 2006] LAVALLE, Steven M.: *Planning Algorithms*. Cambridge : Cambridge University Press, 2006 http://books.google.com/books?hl=en&lr=&id=Clg8SWNMSRAC&oi=fnd&pg=PR11&dq=Planning+Algorithms&ots=gXdqboDmxG&sig=BourFsMJGQ6yCUaKDky3iBdBhLQhttp://ebooks.cambridge.org/ref/id/CBO9780511546877. – ISBN 9780511546877

[Liu u. a. 2007] LIU, H ; MEUSEL, P ; SEITZ, N ; WILLBERG, B ; HIRZINGER, G ; JIN, M H. ; LIU, Y W. ; WEI, R ; XIE, Z W.: The Modular Multisensory DLR-HIT-Hand. In: *Mechanism and Machine Theory* 42 (2007), Mai, Nr. 5, 612–625. http://linkinghub.elsevier.com/retrieve/pii/S0094114X0600098X. – ISSN 0094114X

[Liu u. a. 2008] LIU, Hong ; MEUSEL, P ; HIRZINGER, G ; JIN, Minghe ; LIU, Yiwei ; XIE, Zongwu: The Modular Multisensory DLR-HIT-Hand: Hardware and Software Architecture. In: *Mechatronics, IEEE/ASME Transactions on* 13 (2008), Nr. 4, S. 461–469. – ISSN 1083–4435

[López u. a. 2008] LÓPEZ, Abraham S. ; ZAPATA, René ; LAMA, Maria O.: Sampling-Based Motion Planning : A Survey. In: *Computación y Sistemas* 12 (2008), Nr. 1, S. 5–24

[Lösch 2012] LÖSCH, Martin: *Erkennung menschlicher Aktivitäten zur Belehrung von Robotern*, Karlsruhe Institute for Technology, Diss., 2012. http://digbib.ubka.uni-karlsruhe.de/volltexte/1000035053

[de Luca u. Mattone 2005] LUCA, a. de ; MATTONE, R.: Sensorless Robot Collision Detection and Hybrid Force/Motion Control. In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation* (2005), Nr. April, 999–1004. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1570247. ISBN 0–7803–8914–X

[Luca u. a. 2006] LUCA, Alessandro ; ALBU-SCHAFFER, Alin ; HADDADIN, Sami ; HIRZINGER, Gerd: Collision Detection and Safe Reaction with the DLR-III Lightweight Manipulator Arm. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. Beijing, China : Ieee, Oktober 2006. – ISBN 1–4244–0258–1, 1623–1630

[Mansouri u. Pecora 2013] MANSOURI, I ; PECORA, Federico: A Representation for Spatial Reasoning in Robotic Planning. In: *IROS 2013 Workshop: AI-based Robotics*, 2013

[Martínez Mozos u. a. 2007] MARTÍNEZ MOZOS, Óscar ; TRIEBEL, Rudolph ; JENSFELT, Patric ; ROTTMANN, Axel ; BURGARD, Wolfram: Supervised Semantic Labeling of Places using Information Extracted from Sensor Data. In: *Robotics and Autonomous Systems* 55 (2007), S. 391–402. – ISBN 0921–8890

[Mason 1981] MASON, Matthew T.: Compliance and Force Control for Computer Controlled Manipulators. In: *IEEE Transactions on Systems, Man, and Cybernetics* 11 (1981), Nr. 6, 418–432. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4308708. – ISSN 0018–9472

[McCarthy u. Hayes 1969] MCCARTHY, John ; HAYES, P J.: Some Philosophical Problems from the Standpoint of Artificial Intelligence. In: MELTZER, B (Hrsg.) ; MICHIE, D (Hrsg.) ; SWANN, M (Hrsg.): *Machine Intelligence 4*. Edinburgh University Press, Edinburgh, Scotland, 1969, S. 463–502

[McDermott u. a. 1998] MCDERMOTT, D ; GHALLAB, M ; HOWE, A: PDDL - The Planning Domain Definition Language / University of Washington. Version: 1998. http://www.citeulike.org/user/kira/article/4097279. 1998. – Forschungsbericht

[Micalizio u. Torasso 2007] MICALIZIO, R ; TORASSO, P: On-line Monitoring of Plan Execution: A Distributed Approach. In: *Knowledge-Based Systems* 20 (2007), März, Nr. 2, 134–142. http://linkinghub.elsevier.com/retrieve/pii/S0950705106002036. – ISSN 09507051

[Miller u. Allen 2004] MILLER, AT T. ; ALLEN, PK K.: GraspIt! A Versatile Simulator for Robotic Grasping. In: *IEEE Robotics & Automation Magazine* 11 (2004), Nr. 4, 110–122. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1371616

[Minguez u. a. 2008] MINGUEZ, Javier ; LAMIRAUX, Florent ; LAUMOND, Jean-Paul: Motion Planning and Obstacle Avoidance. In: SICILIANO, B. AND KHATIB, O. (Hrsg.): *Springer Handbook of Robotics*. Heidelberg : Springer Berlin Heidelberg, 2008, Kapitel 35, S. 827–852

[Morinaga u. Kosuge 2003] MORINAGA, S. ; KOSUGE, K.: Collision Detection System for Manipulator Based on Adaptive Impedance Control Law. In: *2003 IEEE International Conference on Robotics and Automation*. Taipei, Taiwan : Ieee, 2003. – ISBN 0–7803–7736–2, 1080–1085

[MPK 2006] *Motion Planning Kit*. http://ai.stanford.edu/ mitul/mpk/, 2006

[Nau u. a. 2003] NAU, D S. ; AU, T C. ; ILGHAMI, O ; KUTER, U ; MURDOCK, J W. ; WU, D ; YAMAN, F: SHOP2: An HTN Planing System. In: *Journal of Artificial Intelligence Research* 20 (2003), S. 379–404

[Nebel u. a. 2013] NEBEL, Bernhard ; DORNHEGE, Christian ; HERTLE, Andreas: How Much Does a Household Robot Need To Know In Order To Tidy Up? In: *AAAI Workshop on Intelligent Robotic Systems*, 2013

[Nilsson 1984] NILSSON, N J.: Shakey the robot / SRI International. 1984 (Tech. Rep. 323). – Forschungsbericht

[Ott u. a. 2006] OTT, C. ; EIBERGER, O. ; FRIEDL, W. ; BAUML, B. ; HILLENBRAND, U. ; BORST, C. ; ALBU-SCHAFFER, A. ; BRUNNER, B. ; HIRSCHMULLER, H. ; KIELHOFER, S. ; KONIETSCHKE, R. ; SUPPA, M. ; WIMBOCK, T. ; ZACHARIAS, F. ; HIRZINGER, G.: A Humanoid Two-Arm System for Dexterous Manipulation. In: *IEEE-RAS International Conference on Humanoid Robots*. Genoa, Italy, 2006. – ISBN 1–4244–0200–X

[Pangercic u. a. 2010] PANGERCIC, Dejan ; TENORTH, Moritz ; JAIN, D. ; BEETZ, M: Combining Perception and Knowledge Processing for Everyday Manipulation. In: *Intelligent Robots and Systems (IROS)*. Taipei, Taiwan, 2010 (Section II), 1065–1071

[Papadias u. Kavouras 1994] PAPADIAS, Dimitris ; KAVOURAS, Marinos: Acquiring, Representing and Processing Spatial Relations. In: *Proceedings of the 6th International Symposium on Spatial Data Handling*. Endinburgh, : Taylor Francis, 1994

[Pardowitz 2007] PARDOWITZ, Michael: *Inkrementelles und interaktives Lernen von Handlungswissen für Haushaltsroboter*, Universität Karlsruhe, Dissertation, 2007. `http://digbib.ubka.uni-karlsruhe.de/volltexte/1000007098`

[Pednault 1986] PEDNAULT, E. P. D.: Formulating Multiagent Dynamic-World Problems in the Classical Planing Framework. In: GEORGEFF, Michale P. (Hrsg.) ; LANSKY, A L. (Hrsg.): *Reasoning about Actions and Plans: Proceedings of the 1986 Workshop*, 1986, S. 47–82

[Pettersson 2005] PETTERSSON, Ola: Execution Monitoring in Robotics: A survey. In: *Robotics and Autonomous Systems* 53 (2005), November, Nr. 2, 73–88. `http://linkinghub.elsevier.com/retrieve/pii/S092188900500134X`. – ISSN 09218890

[Pettersson u. a. 2005] PETTERSSON, Ola ; KARLSSON, Lars ; SAFFIOTTI, Alessandro: Model-Free Execution Monitoring by Learning from Simulation. In: *IEEE International Symposium on Computational Intelligence in Robotics and Automation, 2005. CIRA 2005.* Espoo, Finland : IEEE, 2005. – ISBN 0–7803–9355–4, 505–511

[Prattichizzo u. Trinkle 2008] PRATTICHIZZO, Domenico ; TRINKLE, Jeffrey C.: Grasping. In: SICILIANO, BRUNO, KHATIB, Oussama (Hrsg.): *Springer Handbook of Robotics*. Heidelberg : Springer Berlin Heidelberg, 2008, S. 671–700

[Pronobis u. a. 2009] PRONOBIS, Andrzej ; JENSFELT, Patric ; SJÖÖÖ, Kristoffer ; ZENDER, Hendrik ; GEERT-JAN M KRUIJFF ; MOZOS, Oscar M. ; BURGARD, Wolfram: Semantic Modelling of Space. In: CHRISTENSEN, H.I. (Hrsg.) ; SLOMAN, A. (Hrsg.) ; KRUIJFF, G-J. (Hrsg.) ; WYATT, J. (Hrsg.): *Cognitive Systems*. Heidelberg : Springer, 2009, Kapitel 5, S. 169–225

[Przybylski u. a. 2011] PRZYBYLSKI, M. ; ASFOUR, Tamin ; DILLMANN, Rüdiger: Planning Grasps for Robotic Hands using a Novel Object Representation based on the Medial Axis Transform. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. San Francisco, California, 2011, S. 1781–1788

[Rivlin 1998] RIVLIN, E.: Practical Pushing Planning for Rearrangement Tasks. In: *IEEE Transactions on Robotics and Automation* 14 (1998), Nr. 4, 549–565. `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=704220`. – ISSN 1042296X

[Rogalla 2003] ROGALLA, Oliver: *Abbildung von Benutzerdemonstrationen auf variable Roboterkonfigurationen*. Herdecke, Universität Karlsruhe, Diss., 2003

[Rühl u. a. 2013] RÜHL, Steffen W. ; OBERLÄNDER, J. ; PFOTZER, Lars ; RÖNNAU, A. ; DILLMANN, R.: Rekonfigurationsplanung für das On-Orbit-Servicing modularer Satellitensysteme. In: *62. Deutscher Luft- und Raumfahrtskongress*. Stuttgart, 2013

[Rühl u. a. 2009] RÜHL, Steffen W. ; XUE, Zhixing ; ZOLLNER, JM ; DILLMANN, R.: Integration of a Loop Based and an Event Based Framework for Control of a Bimanual Dextrous

Service Robot. In: *Robotics and Biomimetics (ROBIO), 2009 IEEE International Conference on*, IEEE, 2009, 110–116

[Rus 1992] RUS, D: *Fine Motion Planning for Dexterous Manipulation*, Department of Computer Science, Cornell University, Diss., 1992. http://www.scopus.com/inward/record.url?eid=2-s2.0-0024862290&partnerID=40&md5=92897669943359249c8a4a8d7069d414. – 775–780 S.

[Russell u. Norvig 2003] RUSSELL, Stuart J. ; NORVIG, Peter: *Artificial Intelligence*. 2. ed., in. Upper Saddle River, NJ : Prentice Hall, 2003 (Prentice Hall series in artificial intelligence)

[Sánchez u. Latombe 2003] SÁNCHEZ, G ; LATOMBE, JC: A Single-Query Bi-Directional Probabilistic Roadmap Planner With Lazy Collision Checking. In: *Robotics Research* (2003). http://www.springerlink.com/index/9843341054386hh6.pdf

[Schamburek 2006] SCHAMBUREK, Jan-Ullrich: *Diploma Thesis Manipulation Planning Among Movable Obstacles*, Universität Karlsruhe, Diplomarbeit, 2006

[Schmidt-Rohr 2013] SCHMIDT-ROHR, Sven R.: *Interactive Learning of Probabilistic Decision Making by Service Robots with Multiple Skill Domains*, Karlsruher Instituts für Technologie (KIT), Dissertation, 2013. http://digbib.ubka.uni-karlsruhe.de/volltexte/1000031092

[Schölkopf 1999] SCHÖLKOPF, Bernhard (Hrsg.): *Advances in kernel methods : support vector learning*. Cambridge, Mass. : MIT Press, 1999. – ISBN 0–262–19416–3

[Siciliano, B. and Khatib 2008] SICILIANO, B. AND KHATIB, O. ; SICILIANO, Bruno [. (Hrsg.): *Springer handbook of robotics*. Berlin : Springer, 2008. – ISBN 978–3–540–23957–4

[Siegert u. Bocionek 1996] SIEGERT, Hans-Jürgen ; BOCIONEK, Siegfried: *Robotik: Programmierung intelligenter Roboter*. Springer, 1996

[Simeon u. a. 2002] SIMEON, Thierry ; CORTES, Juan ; SAHBANI, Anis ; LAUMOND, Jean-Paul: A Manipulation Planner for Pick and Place Operations under Continuous Grasps and Placements. In: *IEEE International Conference on Robotics and Automation* Bd. 2, IEEE, 2002, 2022–2027

[Simeon u. a. 2004] SIMEON, Thierry ; CORTES, Juan ; SAHBANI, Anis ; LAUMOND, Jean-Paul: A General Manipulation Task Planner. In: *Algorithmic Foundations of Robotics V* Bd. 7. 2004, S. 311–328

[Simmons 1994] SIMMONS, Ried: Structured Control for Autonomous Robots. In: *IEEE Transactions on Robotics and Automation* 10 (1994), 34–43. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=285583. – ISSN 1042296X

[Simmons u. Apfelbaum 1998] SIMMONS, Ried ; APFELBAUM, D.: A Task Description Language for Robot Control. In: *International Conference on Intelligent Robots and Systems.* Bd. 3. Victoria, B.C., Canada, 1998. – ISBN 0–7803–4465–0

[Sjöö u. Jensfelt 2011] SJÖÖ, K. ; JENSFELT, P.: Learning Spatial Relations from Functional Simulation. In: *International Conference on Intelligent Robots and Systems*. San Francisco : IEEE, September 2011. – ISBN 978–1–61284–456–5, 1513–1519

[Sjöö u. a. 2010] SJÖÖ, Kristoffer ; AYDEMIR, A ; MÖRWALD, Thomas ; JENSFELT, P: Mechanical Support as a Spatial Abstraction for Mobile Robots. In: *International Conference on Intelligent Robots and Systems*. Taipei, Taiwan : IEEE, Oktober 2010. – ISBN 978–1–4244–6674–0, 4894–4900

[Sjöö u. a. 2012] SJÖÖ, Kristoffer ; AYDEMIR, Alper ; JENSFELT, Patric: Topological Spatial Relations for Active Visual Search. In: *Robotics and Autonomous Systems* 60 (2012), September, Nr. 9, 1093–1107. http://linkinghub.elsevier.com/retrieve/pii/S0921889012000851. – ISSN 09218890

[Stilman u. a. 2006] STILMAN, Mike ; NISHIWAKI, Koichi ; KAGAMI, Satoshi ; KUFFNER, J.J.: Planning and Executing Navigation Among Movable Obstacles. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems* Bd. 2. Beijing, China : IEEE, 2006, 820–826

[Stilman u. a. 2007] STILMAN, Mike ; SCHAMBUREK, Jan-Ullrich ; KUFFNER, James ; ASFOUR, Tamim: Manipulation Planning Among Movable Obstacles. In: *IEEE International Conference on Robotics and Automation*, IEEE, April 2007. – ISBN 1–4244–0602–1, 3327–3332

[Stulp u. a. 2009] STULP, F ; FEDRIZZI, A ; BEETZ, M: Learning and performing place-based mobile manipulation. In: *International Conference on Development and Learning* Bd. 2009. Shanghai, China : IEEE, 2009, 1–7

[Sussman 1973] SUSSMAN, G. A. J. ; SUN, Ron (Hrsg.): A Computational Model of Skill Acquisition / MIT. Version: 1973. http://dspace.mit.edu/bitstream/handle/1721.1/6894/AITR-297.pdf?sequence=2. Cambridge University Press, 1973 (297). – Forschungsbericht. – 359–395 S. – ISBN 9780521674102

[Tenorth u. a. 2010] TENORTH, Moritz ; JAIN, Dominik ; BEETZ, Michael: Knowledge Processing for Cognitive Robots. In: *KI-Künstliche Intelligenz* 4 (2010), Nr. 3, 233—240. http://link.springer.com/article/10.1007/s13218-010-0044-0

[Tipler u. a. 2009] TIPLER, P A. ; MOSCA, G ; BASLER, M ; DOHMEN, R ; HEINISCH, C ; SCHLEITZER, A ; ZILLGITT, M: *Physik für Wissenschaftler und Ingenieure*. Spektrum Akademischer Verlag, 2009 (Spektrum Lehrbuch). http://books.google.de/books?id=60HtPgAACAAJ. – ISBN 9783827419453

[Trinkle u. Hunter 1991] TRINKLE, J.C. ; HUNTER, J.J.: A Framework for Planning Dexterous Manipulation. In: *IEEE International Conference on Robotics and Automation*. Sacramento, CA : IEEE, 1991, 1245–1251

[Weld 1994] WELD, Daniel: An Introduction to Least-Commitment Planning. In: *Artificial Intelligence Magazine* (1994), S. 27–61

[Wilfong 1988] WILFONG, G.: Motion Planning in the Presence of Movable Obstacles. In: *ACM Symposium Computational Geometry*, 1988, S. 279–288

[Xue u. a. 2009] XUE, Zhixing ; KASPER, Alexander ; ZÖLLNER, J. M. ; DILLMANN, Rüdiger: An Automatic Grasp Planning System for Service Robots. In: *International Conference on Advanced Robotics*. Munich. Germany : IEEE, 2009, 1–6

[Xue u. a. 2010] XUE, Zhixing ; RÜHL, Steffen W. ; HERMANN, Andreas ; KERSCHER, Thilo ; DILLMANN, Rüdiger: Autonomous Grasp and Manipulation Planning using a ToF Camera. In: *Robotics and Autonomous Systems* 60 (2010), August, Nr. 3, 387–395. http://linkinghub.elsevier.com/retrieve/pii/S0921889011001448. – ISSN 09218890

[Xue u. a. 2012] XUE, Zhixing ; RÜHL, Steffen W. ; ZÖLLNER, J M. ; DILLMANN, Rüdiger: An Automatic Grasp Planning System for Multi-Fingered Robotic Hands. In: PRASSLER, Erwin (Hrsg.) ; BISCHOFF, Rainer (Hrsg.) ; BURGARD, Wolfram (Hrsg.) ; HASCHKE, Robert (Hrsg.) ; HÄGELE, Martin (Hrsg.) ; LAWITZKY, Gisbert (Hrsg.) ; NEBEL, Bernhard (Hrsg.) ; PLÖGER, Paul (Hrsg.) ; REISER, Ulrich (Hrsg.) ; ZÖLLNER, Marius (Hrsg.): *Towards Service Robots for Everyday Environments*. Berlin : Springer, 2012 (Springer Tracts in Advanced Robotics - STAR 76), 2012. – ISBN 978–3–642–25115–3, S. 391–402

[Xue u. a. 2008a] XUE, Zhixing ; SCHMIDT, Michael ; ZÖLLNER, J. M. ; DILLMANN, Rüdiger: Internal Force Computation of Grasped Object Using Joint Torques. In: *2008 SICE Annual Conference*. Tokyo, Japan : IEEE, August 2008. – ISBN 978–4–907764–30–2, 2795–2800

[Xue u. a. 2007] XUE, Zhixing ; ZOELLNER, J. M. ; DILLMANN, Ruediger: Grasp Planning: Find the Contact Points. In: *IEEE International Conference on Robotics and Biomimetics (ROBIO)*. Sanya, China : IEEE, Dezember 2007. – ISBN 978–1–4244–1761–2, 835–840

[Xue u. a. 2008b] XUE, Zhixing ; ZÖLLNER, J. M. ; DILLMANN, Rüdiger: Planning Regrasp Operations for a Multifingered Robotic Hand. In: *Automation Science and Engineering* Bd. M. Arlington, VA, 2008, 778 – 783

[Yoshikawa 1985] YOSHIKAWA, T: Manipulability of Robotic Mechanisms. In: *International Journal of Robotics Research* 4 (1985), April, S. 3–9

[Zacharias u. a. 2006] ZACHARIAS, F ; BORST, C ; HIRZINGER, G: Bridging the Gap between Task Planning and Path Planning. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems,*. Beijing, China, 2006, S. 4490–4495

[Zacharias u. a. 2007] ZACHARIAS, Franziska ; BORST, Christoph ; HIRZINGER, Gerd: Capturing Robot Workspace Structure: Representing Robot Capabilities. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. San Diego, California, USA, 2007 (7), 3229–3236

[Zöllner 2006] ZÖLLNER, Raoul D.: *Erlernen zweihändiger feinmotorischer Handhabungen*. Waabs, Universität Karlsruhe, Diss., 2006. `http://www.gbv.de/dms/ilmenau/toc/52690786X.PDF`

Eigene Publikationen

[Grundmann u. a. 2008] GRUNDMANN, Thilo ; EIDENBERGER, Robert ; ZOELLNER, RD Raoul D. ; XUE, Zhixing ; RUEHL, Steffen ; ZOELLNER, J M. ; DILLMANN, Ruediger ; KUEHNLE, Jens ; VERL, Alexander: Integration of 6d object localization and obstacle detection for collision free robotic manipulation. In: *2008 IEEESICE International Symposium on System Integration (2008)*. Nagoya, 2008

[Herbst u. a. 2013a] HERBST, Uwe ; RÜHL, Steffen ; HERMANN, Andreas ; XUE, Zhixing ; BENGLER, Klaus: Ergonomic rating of interaction technologies for a mobile robot system. In: *HCI International*. Las Vegas, USA, 2013

[Herbst u. a. 2013b] HERBST, Uwe ; RÜHL, STEFFEN WILHELM HERMANN, Andreas ; XUE, Zhixing ; BENGLER, Klaus: Ergonomic 6D Interaction Technologies for a Flexible and Transportable Robot System: A Comparison. In: *Analysis, Design, and Evaluation of Human-Machine Systems*. Las Vegas, United States of America, 2013, S. 58–63

[Hermann u. a. 2011] HERMANN, Andreas ; XUE, Zhixing ; RUEHL, Steffen W. ; DILLMANN, Rüdiger: Hardware and software architecture for a bimanual mobile manipulator in industrial applications. In: *Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on*. Phuket, Thailand : IEEE, 2011

[Jäkel u. a. 2012a] JÄKEL, Rainer ; RÜHL, W S. ; SCHMIDT-ROHR, Sven R. ; LÖSCH, Martin ; XUE, Zhixing ; DILLMANN, Rüdiger: Layered Programming by Demonstration and Planning for Autonomous Robot Manipulation. Version: 2012. http://dx.doi.org/10.1007/978-3-642-29041-1_1. In: SICILIANO, Bruno (Hrsg.): *Springer Tracts in Advanced Robotics* Bd. 80. Springer Berlin Heidelberg, 2012. – DOI 10.1007/978–3–642–29041–1_1. – ISBN 9783642290404, 1–57

[Jäkel u. a. 2012b] JÄKEL, Rainer ; SCHMIDT-ROHR, SvenR. ; RÜHL, SteffenW. ; KASPER, Alexander ; XUE, Zhixing ; DILLMANN, Rüdiger: Learning of Planning Models for Dexterous Manipulation Based on Human Demonstrations. In: *International Journal of Social Robotics* 4 (2012), Nr. 4, 437–448. http://dx.doi.org/10.1007/s12369-012-0162-y. – DOI 10.1007/s12369–012–0162–y. – ISSN 1875–4791

[Kühnle u. a. 2009] KÜHNLE, Jens ; VERL, Alexander ; XUE, Zhixing ; RÜHL, Steffen W. ; ZÖLLNER, J.M. M. ; DILLMANN, Ruediger ; GRUNDMANN, Thilo ; EIDENBERGER, Robert ; ZÖLLNER, R.D. Raoul D. ; DILLMAN, Rüdiger: 6D Object Localization and Obstacle Detection for Collision-Free Manipulation with a Mobile Service Robot. In: *14th International Conference on Advanced Robotics (ICAR)*. Munich : IEEE, 2009, 1–6

[Rühl u. a. 2013a] RÜHL, S. ; OBERLÄNDER, J. ; PFOTZER, Lars ; RÖNNAU, A. ; DILLMANN, R.: Rekonfigurationsplanung für das On-Orbit-Servicing modularer Satellitensysteme. In: *62. Deutscher Luft- und Raumfahrtskongress*. Stuttgart, 2013

[Rühl u. a. 2013b] RÜHL, S. ; PFOTZER, L. ; OBERLÄNDER, J. ; RÖNNAU, A. ; DILLMANN, R.: Aktorauslegung des Servicer-Satelliten beim On-Orbit Servicing modularer Satelliten. In: *62. Deutscher Luft- und Raumfahrtskongress*. Stuttgart, 2013

[Rühl u. a. 2008] RÜHL, Steffen W. ; GRINBERG, Michael ; WILLERSINN, Dieter: Empirical evaluation of motion models for a side-looking driver assistance system. In: *Proceedings of the 5th International Workshop on Intelligent Transportation (WIT 2008), Hamburg*, 2008, S. 19–24

[Rühl u. a. 2011a] RÜHL, Steffen W. ; HERMANN, Andreas ; XUE, Zhixing ; KERSCHER, Thilo ; DILLMANN, Ruediger: Generating a Symbolic Scene Description for Robot Manipulation Using Physics Simulation. In: *Multibody Dynamics*. Brussels, Belgium, 2011

[Rühl u. a. 2011b] RÜHL, Steffen W. ; HERMANN, Andreas ; XUE, Zhixing ; KERSCHER, Thilo ; DILLMANN, Ruediger: Graspability: A Description of Work Surfaces For Planning of Robot Manipulation Sequences. In: *2011 IEEE International Conference on Robotics and Automation*. Shanghai, China : IEEE, 2011

[Rühl u. a. 2014] RÜHL, Steffen W. ; OBERLAENDER, Jan ; ROENNAU, Arne ; DILLMANN, Ruediger: Using A Robot Workspace Description for Planning On-Orbit Servicing Tasks on Modular Satellites. In: *12th International Symposium on Artificial Intelligence, Robotics and Automation in Space - i-SAIRAS 2014*. Montreal, Kanada, 2014

[Rühl u. a. 2012] RÜHL, Steffen W. ; XUE, Zhixing ; DILLMANN, Rüdiger: Monitoring of manipulation activities for a service robot using supervised learning. In: *2012 IEEE International Conference on Robotics and Automation*, IEEE, Mai 2012. – ISBN 978–1–4673–1405–3, 930–935

[Rühl u. a. 2010] RÜHL, Steffen W. ; XUE, Zhixing ; KERSCHER, Thilo ; DILLMANN, Ruediger: Towards Automatic Manipulation Action Planning for Service Robots. In: *33rd Annual German Conference on Artificial Intelligence (KI 2010)*. Karlsruhe, September 2010

[Rühl u. a. 2009] RÜHL, Steffen W. ; XUE, Zhixing ; ZOLLNER, JM ; DILLMANN, R.: Integration of a loop based and an event based framework for control of a bimanual dextrous service robot. In: *Robotics and Biomimetics (ROBIO), 2009 IEEE International Conference on*, IEEE, 2009, 110–116

[Xue u. a. 2012a] XUE, Zhixing ; KUEHNLE, Jens ; RUEHL, Steffen W. ; DILLMANN, Ruediger: Motion Planning using a ToF Camera for Manipulation in Cluttered Environments. Version: 2012. http://dx.doi.org/10.1007/978-3-642-25116-0. In: PRASSLER, Erwin (Hrsg.) ; BISCHOFF, Rainer (Hrsg.) ; BURGARD, Wolfram (Hrsg.) ; HASCHKE, Robert (Hrsg.) ; HÄGELE, Martin (Hrsg.) ; LAWITZKY, Gisbert (Hrsg.) ; NEBEL, Bernhard (Hrsg.) ; PLÖGER, Paul (Hrsg.) ; REISER, Ulrich (Hrsg.) ; ZÖLLNER, Marius (Hrsg.): *Towards Service Robots for Everyday Environments : Recent Advances in Designing Service Robots for Complex Tasks in Everyday Environments*. Berlin : Springer, 2012 (Springer Tracts in Advanced Robotics - STAR 76), 2012. – DOI 10.1007/978–3–642–25116–0. – ISBN 978–3–642–25115–3, S. 567–577

[Xue u. a. 2011] XUE, Zhixing ; RUEHL, Steffen ; HERMANN, Andreas ; KERSCHER, Thilo ; DILLMANN, Ruediger: An autonomous ice-cream serving robot. In: *2011 IEEE International Conference on Robotics and Automation*, IEEE, Mai 2011. – ISBN 978–1–61284–386–5, 3451–3452

[Xue u. a. 2010] XUE, Zhixing ; RÜHL, Steffen W. ; HERMANN, Andreas ; KERSCHER, Thilo ; DILLMANN, Ruediger: Autonomous Grasp and Manipulation Planning using a ToF Camera. In: *Robotics and Autonomous Systems* Bd. 60. Shanghai, China, August 2010. – ISSN 09218890, 387–395

[Xue u. a. 2012b] XUE, Zhixing ; RÜHL, Steffen W. ; ZÖLLNER, J M. ; DILLMANN, Rüdiger: An Automatic Grasp Planning System for Multi-Fingered Robotic Hands. Version: 2012. http://dx.doi.org/10.1007/978-3-642-25116-0_26. In: PRASSLER, Erwin (Hrsg.) ; BISCHOFF, Rainer (Hrsg.) ; BURGARD, Wolfram (Hrsg.) ; HASCHKE, Robert (Hrsg.) ; HÄGELE, Martin (Hrsg.) ; LAWITZKY, Gisbert (Hrsg.) ; NEBEL, Bernhard (Hrsg.) ; PLÖGER, Paul (Hrsg.) ; REISER, Ulrich (Hrsg.) ; ZÖLLNER, Marius (Hrsg.): *Towards Service Robots for Everyday Environments*. Berlin : Springer, 2012 (Springer Tracts in Advanced Robotics - STAR 76), 2012. – DOI 10.1007/978–3–642–25116–0_26. – ISBN 978–3–642–25115–3, S. 391–402